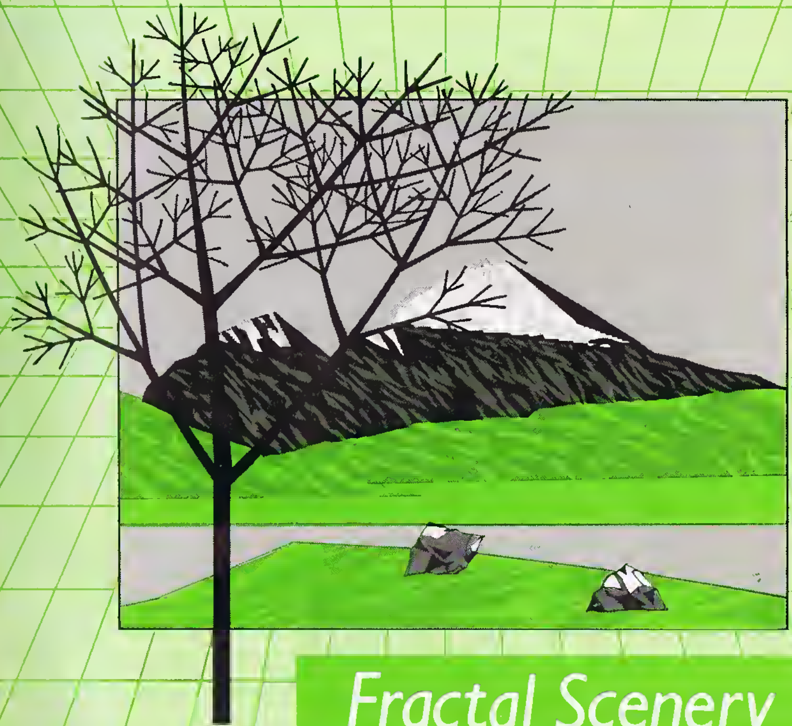
Volume 2
Issue 3

January/
February
1989

Price £1.20

RISC USER



Fractal Scenery

THE MAGAZINE AND SUPPORT GROUP EXCLUSIVELY FOR USERS OF THE ARCHIMEDES

Volume 2 Issue 3

January/February 1989

RISC USER

CONTENTS

RISC User is published by BEEBUG Ltd.

Co-Editors: Mike Williams, Dr Lee Calcraft
Assistant Editor: Kristina Lucas
Technical Editor: David Spencer
Production assistant: Sheila Stoneman
Advertising: Sarah Shrive
Subscriptions: Mandy Mileham

BEEBUG, Dolphin Place, Holywell Hill, St.Albans, Herts AL1 1EX. Tel. St.Albans (0727) 40303

All rights reserved. No part of this publication may be reproduced without prior written permission of the Publisher. The Publisher cannot accept any responsibility whatsoever for errors in articles, programs, or advertisements published. The opinions expressed on the pages of this journal are those of the authors and do not necessarily represent those of the Publisher, BEEBUG Limited.

BEEBUG Ltd (c) 1989

Printed by Arlon House

EDITORIAL

As you will see from this issue of RISC User, we are beginning to focus our attention on the new Archimedes operating system called RISC OS, which will be available in April of this year. We feel very enthusiastic about this development, particularly since we had a first chance to see and use the new system at the PC Show last September, and the greater access which Acorn has allowed us since. The new system together with the vastly improved desktop is a major advance on what has been available so far, and in our opinion, the Archimedes with RISC OS certainly surpasses the Apple Macintosh which we use to produce RISC User. It is more colourful, more powerful, and on first impressions just as user friendly.

Neither we, nor of course Acorn, see RISC OS as an on-going alternative to Arthur. RISC OS is *the* operating system for the Archimedes in the future, and once the new system is readily available, we shall only be publishing and supporting programs in RISC User which function under RISC OS (though they may still work with Arthur too). We feel that this move is important for all Archimedes users and for Acorn. The upgrade represents excellent value at a price which will barely cover the cost to Acorn of the upgrade. We urge all readers to upgrade to RISC OS, and get the most out of their Archimedes (you can reserve your copy of RISC OS for £33.35 inc. VAT + £3 p&p through BEEBUG, for immediate delivery when available).

We are well aware that readers cannot experience the delights of RISC OS now, but we feel that it is important that we prepare the way, starting this month with two short articles. One gives an overview of the new desktop, while the other gets to the heart of RISC OS by taking a look at multi-tasking. By April we expect that our first programs specifically for RISC OS will have appeared. Of course, many, if not all, of the programs already published will still work under RISC OS, or will need very little alteration to do so.

This January/February issue of RISC User covers two months. The next issue will be that for March.

*This month's telesoftware password is **halfpenny.***
(see BEEBUG pages on Micronet)

205 OR NOT 205?

It has been rumoured that Acorn is working on a cut-down Archimedes, currently called the 205. It is claimed the new machine is designed to fill a gap in the primary school education market, and to compete against the Amiga and Atari ST. It is thought that the 205 will use the same four chips that form the heart of the current machines, although it will be limited to 2Mb of RAM. A suggested price for the 205 with a single disc drive and 1Mb of RAM is £599, and to achieve this it would probably be necessary to sacrifice most of the expansion capabilities of the Archimedes. There is also no indication of the operating system that would be supplied, but the cheapest option would seem to be a modified version of either the current Arthur, or more probably, RISC OS.

UNIX IS HERE

Well almost! The long awaited ARM based UNIX system has been demonstrated to the press and will be shown publicly for the first time at the educational show BETT '89 at the end of January. The official launch will be at the Which Computer show at the NEC a few days later. The system, costing around £4000 (ex. VAT), is called the R140, and is very similar to the Archimedes 440 in terms of hardware, the only real difference being a 50Mb hard disc in the R140. The operating system of the R140 is the standard Berkeley 4.3 implementation of UNIX with extensions to bring the specification up to that of system 5 UNIX. Also supplied is *X-Desktop* from IXI, which is a WIMP system and Desktop Manager to allow easier access to the UNIX commands. The bundled software includes productivity software from Uniplex and compilers for C, Fortran and Pascal.

It will be possible to read ADFS, MS-DOS and Xenix discs on the floppy drive, and standard Sun Microsystems software will allow networking through an optional Ethernet card. Future add-ons will include the promised floating point co-processor and a SCSI interface to allow the connection of larger hard discs, tape streamers, etc.

ACORN COMPILERS

Acorn has released new versions of its C, ISO Pascal, and Fortran 77 compilers. All the new versions include various enhancements and improvements designed to both extend the facilities offered, and also to improve performance. There are new user manuals for each compiler, the ones for C and Fortran being completely new guides, while the one for Pascal is a supplement to the existing guide. The new Fortran compiler now offers some access to operating system routines, while the new version of C includes features to make it easier to port programs for use on UNIX PCC standard compilers. Additionally, the C compiler can support a 'shared library' to make RISC OS applications more compact. Anybody wishing to use the Acorn Symbolic Debugger from the Software Developer's Toolbox (see review in this issue) will need the new compilers for this to work correctly.

The price of the compilers is the same as for the old versions (£108.16 inc. VAT to RISC User members). However, until 1st April 1989 owners of the original versions can obtain the new releases by sending £30 inc. VAT along with their original disc (only) to Acorn Direct, Studland Road, Kingsthorpe, Northampton NN2 6NA.

THROUGH THE ARCHED WINDOW

Archway from Simtron is a new package designed to make it easier to use the Archimedes' window system. Basically, *Archway* sits between the WIMP Manager and the user, and allows window based programs to be written without needing a detailed knowledge of the WIMP Manager.

Archway is supplied on four discs with a 300 page manual and costs £79.95 (inc. VAT). More details from: Simtron, 4 Clarence Drive, East Grinstead, West Sussex RH19 4RX, tel. (0342) 28188.

ROLL ON REPTON

Repton 3, the thinking man's game, is now available for the Archimedes. The new version of the classic game from Superior Software is written entirely in ARM machine code, although it is only slightly enhanced over the BBC version. The *Repton 3* disc contains not only the original game, but also 120 additional game screens. The Archimedes version of *Repton 3* costs £19.95 inc. VAT, and is available from Archimedes dealers, including BEEBUG. RU

Archimedes Visuals

Using fractal geometry to create sequences of trees and complete landscapes.

The Mandlebrot Set, featured in RISC User some months ago is based on a branch of mathematics called Fractal Geometry. In our Mandelbrot program, the colour of any pixel on the screen was determined by the number of iterations of a simple formula before convergence occurs. The result is a regular, but infinitely detailed two-dimensional landscape.

In creating fractal trees and landscapes, a slightly different technique is applied. Here a shape is repeatedly *split*, and with each split, a small, fractional distortion is introduced. In the case of a tree, a branch splits into two (or more) smaller branches, and these each split further, and so on. In the case of a landscape, a triangular (or other) area is bisected, the two areas are shaded in different colours, and then each is further bisected, until the required limit is reached. At each bisection, a small distortion (random or otherwise) is introduced, to give a more natural appearance to the object, which in this case might be a mountain, a hill, or a stretch of grass.

Fractal Tree
by Andrew Brooks

The first program uses a random factor with each recursion to draw a sequence of trees. The program creates a completely random tree, and waits for the space bar to be pressed before drawing another. The trunk of the tree appears in brown. The base of each branch is light green, and with every level of recursion the green darkens. The effect is good, and the procedure could well be used to create scenery for any purpose.

Notes:

PROCtrunk(x,y) draws a simple trunk from x,0 to x,y.

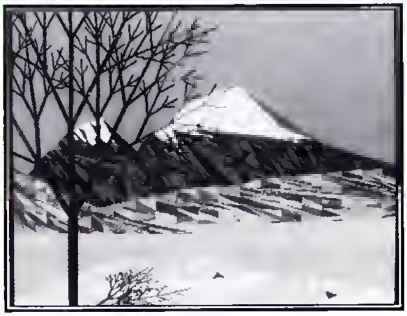
PROCtree(x,y,depth,angle) draws branches from a point x,y. The most complex tree is drawn when depth=1 (it must not be less than 1). Set angle=90 for a broadly symmetrical span of branches.

```
   10 REM           >Tree
   20 REM Program   Fractal Tree
   30 REM Version   A 1.0
   40 REM Author    Andrew Brooks
   50 REM RISC User Jan/Feb 1989
   60 REM Program   Subject to Copyright
   70 :
   80 maxdepth=10:maxlength=150
   90 maxsplits=3:anglerange=90
  100 depth%=1:angle%=90
  110 X%=640:Y%=200
  120 MODE12:OFF:PROCcolours
  130 REPEAT
  140  CLS:PROCtrunk(X%,Y%)
  150  PROCtree(X%,Y%,depth%,angle%)
  160  PRINT"Press space to run again."
  170 UNTIL GET<>32
  180 -----------------------------------
  190 DEF PROCtree(x%,y%,depth%,angle%)
200 IF depth%<=maxdepth THEN
  210  LOCAL I%,num%,newang%,len%,x1%,y1
%,width%
  220  num%=RND(maxsplits)
  230  width%=30/(depth%+1)
  240  FOR I%=1 TO num%
  250   newang%=angle%+FNturn
  260   len%=RND(maxlength/(LNdepth%+1))
  270   x1%=x%+len%*COSRADnewang%
  280   y1%=y%+len%*SINRADnewang%
  290   GCOL0,depth%
  300   MOVE x%,y%:MOVE x%+width%,y%
  310   PLOT117,x1%,y1%
  320   PROCtree(x1%-width%/2,y1%,depth%
+1,newang%)
  330  NEXT
  340 ENDIF
  350 ENDPROC
  360 :
```

```
370 DEF FNturn
380 =RND(anglerange)-anglerange/2
390 :
400 DEF PROCtrunk(x%,y%)
410 GCOL0:MOVE x%,0:MOVEx%+20,0
420 PLOT117,x%+20,y%
430 ENDPROC
440 :
450 DEF PROCcolours
460 FOR I%=1 TO maxdepth
470 COLOUR I%,0,(15-I%)*16,0
480 NEXT
490 COLOUR15,&10,&C0,&F0
500 COLOUR 0,&A0,&80,&50
510 COLOUR 1,&A0,&C0,&60
520 COLOUR128+15
530 ENDPROC
```

Fractal Scenery
by John Greening

This slightly longer program uses Fractal Geometry to generate a complete scene, which includes snow-topped mountains, hills, vales, water, sunlit rocks and a leafless tree and its shadow. The central procedure PROCfractal is used for drawing the complete landscape, except for the tree (which uses the separate procedure PROCtree).



You can alter the landscape very easily, by changing the parameters used. PROCfractal takes 8 parameters. The first 6 are the three co-ordinate pairs of the triangular border of the object (all objects are treated as combinations of triangles). The seventh indicates the fineness of recursion, smaller values giving a finer structure. A value of 40 is used for the mountains, and 80 for the foreground. Finally comes the colour number of the object (the program uses the four tints to create shading effects).

PROCfractal works by splitting up triangular areas, repeatedly bisecting sides, while at the same time distorting the difference between the ordinates of the bisected points by adding 0.2 times the difference between the ordinates of the bisected sides. Random factors have not been used for the landscape, though you may like to experiment with this.

Finally the variable *snowline* can be set before the mountains are drawn. If you do not want snow to appear, simply make *snowline* exceed the largest ordinate of the feature being drawn.

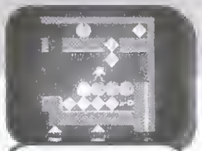
This program contains some powerful procedures, and is well worth experimenting with. By making small adjustments to the parameters used, the scenery can be completely redesigned.

```
   10 REM          >FracScene
   20 REM Program  Fractal Scenery
   30 REM Version  1.2
   40 REM Author   John Greening
   50 REM RISC User Jan/Feb 1989
   60 REM Program  Subject to copyright
   70 :
   80 ON ERROR:REPORT:PRINT" at line ";E
RL:END
   90 MODE 15
  100 PROCskywater
  110 PROCgrass
  120 snowline=550
  130 PROCmountains
  140 PROChills
  150 PROCforeground
  160 PROCrocks
  170 MOVE200,-4
  180 PROCtree(PI/20,76,1,6,9,9,0)
  190 PROCtree(PI/2,300,1,6,16,1,64)
  200 END
  210 :
  220 DEFPROCfractal(x1,y1,x2,y2,x3,y3,l
en%,col%)
  230 IF ABS(x2-x1)>=len% THEN
  240 LOCAL x4,y4,x5,y5,x6,y6
  250 x4=(x1+x2)/2
  260 y4=(y1+y2)/2+0.2*(y1-y2)
  270 x5=(x2+x3)/2
  280 y5=(y2+y3)/2+0.2*(y2-y3)
  290 x6=(x3+x1)/2
  300 y6=(y3+y1)/2+0.2*(y3-y1)
  310 MOVE x3,y3:MOVE x6,y6
  320 GCOL col% TINT 64*(RND(4)-1)
  330 PROCsnowtint:PLOT 85,x5,y5
  340 GCOL col% TINT 64*(RND(4)-1)
  350 PROCsnowtint:PLOT 85,x4,y4
  360 GCOL col% TINT 64*(RND(4)-1)
  370 PROCsnowtint:PLOT 85,x2,y2
  380 MOVE x4,y4:MOVE x6,y6
  390 GCOL col% TINT 64*(RND(4)-1)
  400 PROCsnowtint:PLOT 85,x1,y1
  410 PROCfractal(x1,y1,x4,y4,x6,y6,len%
,col%)
  420 PROCfractal(x4,y4,x2,y2,x5,y5,len%
,col%)
  430 PROCfractal(x6,y6,x5,y5,x3,y3,len%
,col%)
  440 PROCfractal(x5,y5,x6,y6,x4,y4,len%
,col%)
  450 ENDIF
  460 ENDPROC
  470 :
  480 DEFPROCsnowtint
  490 IF y1>snowline GCOL 63 TINT 64*(RN
D(2)+1)
  500 ENDPROC
  510 :
  520 DEFPROCtree(angle,height%,branches
%,depth%,girth%,col%,tint%)
  530 LOCAL I%,X%,Y%
  540 IF depth%<>0 THEN
  550 X%=height%*COS(angle)
  560 Y%=height%*SIN(angle)
  570 GCOL col% TINT tint%
  580 FOR I%=1 TO branches%
  590 PLOT1,girth%,0:PLOT1,X%,Y%:PLOT81,
-X%-2*girth%,-Y%:PLOT81,X%,Y%:PLOT1,girt
h%,0
  600 PROCtree(angle+RAD(25+RND(10)),hei
ght%/1.5,branches%,depth%-1,girth%/1.5,c
ol%,tint%)
  610 PROCtree(angle+RAD(-5+RND(10)),hei
ght%/1.05,branches%,depth%-1,girth%/1.5,
col%,tint%)
  620 PROCtree(angle-RAD(25+RND(10)),hei
ght%/1.5,branches%,depth%-1,girth%/1.5,c
ol%,tint%)
  630 PLOT 0,-X%,-Y%
  640 NEXT
  650 ENDIF
  660 ENDPROC
  670 :
  680 DEFPROCskywater
  690 VDU 24,0;0;1279;230;
  700 VDU28,0,24,79,0
  710 COLOUR 128+48 TINT 192:CLS
  720 GCOL 128+56 TINT128:CLG
  730 VDU26
  740 ENDPROC
  750 :
  760 DEFPROChills
  770 PROCfractal(200,280,1300,300,1300,
450,80,9)
  780 PROCfractal(0,250,600,250,0,450,40
,9)
  790 ENDPROC
  800 :
  810 DEFPROCgrass
  820 GCOL 29 TINT 0
  830 MOVE0,300:MOVE 1280,300
  840 PLOT 85,0,200:PLOT 85,1280,200
  850 ENDPROC
  860 :
  870 DEFPROCrocks
  880 GCOL 9 TINT 0
  890 MOVE 640,104:MOVE 640,150:PLOT 85,
700,108
  900 MOVE 1050,32:MOVE 1032,60:PLOT 85,
1080,44
  910 PROCfractal(550,100,640,100,700,17
5,40,27)
  920 PROCfractal(550,100,700,175,600,20
0,40,27)
  930 PROCfractal(900,30,1050,30,950,100
,40,27)
  940 PROCfractal(1050,30,950,100,975,12
0,40,27)
  950 ENDPROC
  960 :
  970 DEFPROCforeground
  980 PROCfractal(-200,0,350,160,1400,0,
80,29)
  990 ENDPROC
 1000 :
 1010 DEFPROCmountains
 1020 PROCfractal(-10,290,700,350,300,60
0,40,36)
 1030 PROCfractal(300,300,1300,450,750,7
00,40,36)
 1040 ENDPROC
```

Archimedes

PRIZE COMPETITION
Can You Design the Best Archimedes Repton 3 Screen ?

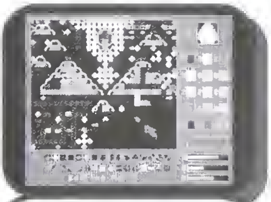
PRIZE COMPETITION
Can You Become the Archimedes Repton 3 World Champion ?


Repton 3

Around the World in 40 Screens

The Life of Repton

Repton Thru Time


The Screen Editor
Redesigning an Egyptian Repton screen

The Character Editor
Rory McRepton brandishing his claymore!

The Repton series of games is the most successful software ever published for the BBC Micro/Electron computers and has aptly been described as "the thinking man's arcade game."

Now, a massive enhanced version is available for the Archimedes, comprising four complete games

- **Repton 3**
- **The Life of Repton**
- **Around the World in 40 Screens**
- **Repton Thru Time**

Enhanced game features include

- A choice of four tunes
- A position Save/Recall facility

In addition, the software includes a sophisticated **Screen Editor** and a **Character Editor** with a choice of 16 colours from a total of 4096 hues. An enhanced feature is that you can design new screens on the Screen Editor and immediately test them by entering the game section.

[illegible]

SUPERIOR SOFTWARE Limited

ACORNSOFT

(Acornsoft is a registered trademark of Acorn Computers Ltd. Superior Software Ltd is a registered user.)

Dept AR1, Regent House, Skinner Lane, Leeds LS7 1AX. Telephone:(0532) 459453

Please make all cheques payable to "Superior Software Ltd"

Access VISA

24 HOUR TELEPHONE ANSWERING SERVICE FOR ORDERS

OUR GUARANTEE
- All mail orders are despatched within 24 hours by first-class post
- Postage and packing is free
- Faulty cassettes and discs will be replaced immediately

This does not affect your statutory rights

ARC FULL-SCREEN ZOOM AND PAN

Use this short relocatable module by Mike Ironmonger to create real-time zoom and pan effects in the 256 colour modes.

This program requires 160K of screen RAM.

The first of the two accompanying programs creates a relocatable module which provides a single star command giving access to a variety of zoom effects. Because of the speed of the native ARM, these effects appear to be instantaneous in their execution, even though they involve altering the whole contents of the screen many times a second.

The second short program calls the module, and puts it through its paces, as we shall see in a moment.

First of all, carefully type in listing 1, and save it to disc before running it. When it is run, it will assemble the module, and save it to disc under the name:

```
Zoomer256
```

At the same time, it will install it in the RMA, and will invoke a list of all the modules in your machine. *Zoomer256* should be the last. You will also see displayed the response to:

```
*HELP ZOOM
```

This gives the command syntax as follows:

```
*ZOOM [<zoom number> [<X> [<Y> [<source bank> [<destination bank>]]]]]
```

Square brackets indicate that the enclosed item or items may be optionally omitted. The default values for the parameters are 0, 0, 0, 1 and 2 respectively.

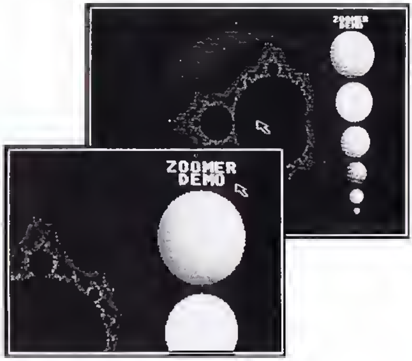
There are seven possible zoom numbers from zero to six, giving the following magnifications: 1, 2, 4, 8, 16, 32, 64. The X and Y co-ordinates specify the top left co-ordinates of the area which is zoomed. Finally the source and destination bank numbers are screen bank numbers of the source and destination screen (normally 1 and 2 respectively).

Zoom works by magnifying a part of the screen held in the source bank, and placing it into the destination bank. Before using the program, you will need enough screen RAM for two screen banks (i.e. 160K for the two mode 13 screens).

To take an example, if you issue:

```
*ZOOM 2,200,500,1,2
```

a copy of the screen in bank 1 will appear in bank 2, magnified by a factor of 4. The top left-hand corner of the zoomed image will correspond to co-ordinates 200,500 on the original.



Listing 1

```
 10 REM              >ZoomSrc
 20 REM Program     Pixel Zoom Module
 30 REM Version     A 1.00
 40 REM Author      Mike Ironmonger
 50 REM RISC User Jan/Feb 1989
 60 REM Program     Subject to Copyright
 70 :
 80 MODE 0
 90 DIM code% 1000
100 sp=13:link=14
110 FOR A%=4 TO 6 STEP 2
120 O%=code%:P%=0
130 [OPT A%
140 .module_header
150 EQUD 0:EQUD 0
160 EQUD 0:EQUD 0
170 EQUD title
180 EQUD help
190 EQUD commands
200 :
210 .title
220 EQUS "Zoomer256"+CHR$0
```

```
  230 .help
  240 EQUS "Zoomer256"+CHR$9+"1.00 ("+MI
D$(TIME$,5,11)+")"+CHR$0
  250 :
  260 .commands
  270 EQUS "Zoom"+CHR$0:ALIGN
  280 EQUD zoom_command
  290 EQUB 0:EQUB 0
  300 EQUB 5:EQUB 0
  310 EQUD zoom_syntax
  320 EQUD zoom_help
  330 EQUB 0
  340 :
  350 .zoom_help
  360 EQUS "*Zoom zooms a 256-colour scr
een"+CHR$13
  370 .zoom_syntax
  380 EQUS "Syntax: *Zoom [<zoom number>
 [<X> [<Y> [<source bank> [<destination
bank>]]]]]"+CHR$0:ALIGN
  390 :
  400 .same_bank_error
  410 EQUD 0:EQUS "Can't zoom onto the s
ame screen bank"+CHR$0:ALIGN
  420 .insufficient_mem_error
  430 EQUD 0:EQUS "Insufficient memory f
or screen bank"+CHR$0:ALIGN
  440 .wrong_mode_error
  450 EQUD 0:EQUS "*Zoom is for 256-colo
ur modes only"+CHR$0:ALIGN
  460 .bad_zoom_number_error
  470 EQUD 0:EQUS "Zoom number must be 0
 to 6"+CHR$0:ALIGN
  480 :
  490 .zoom_command
  500 STMFD (sp)!,{link}
  510 MOV   R2,#0:MOV   R3,#0
  520 MOV   R4,#0:MOV   R5,#1
  530 MOV   R6,#2
  540 STMDB (sp)!,{R2-R6}
  550 :
  560 .read_numbers
  570 MOV   R4,sp:MOV   R3,R1
  580 MOV   R1,R0
  590 MOV   R0,#&8000000A
  600 .next_number
  610 SUBS  R3,R3,#1
  620 SWIPL "OS_ReadUnsigned"
  630 STRPL R2,[R4],#4
  640 BPL   next_number
  650 LDMIA (sp)!,{R3-R7}
  660 :
  670 .check_xco
  680 MOV   R0,#1280
  690 ADD   R0,R4,R0,LSR R3
  700 RSBS  R0,R0,#1280
  710 ADDMI R4,R4,R0
  720 :
  730 .check_yco
  740 MOV   R5,R5,LSR #2
  750 MOV   R0,#256
  760 ADD   R0,R5,R0,LSR R3
  770 RSBS  R0,R0,#256
  780 ADDMI R5,R5,R0
  790 :
  800 .check_mode
  810 MOV R0,#135:SWI "OS_Byte"
  820 TEQ   R2,#10
  830 TEQNE R2,#13
  840 MOVEQ R10,#320
  850 MOVEQ R4,R4,LSR #2
  860 BEQ   check_zoom_number
  870 TEQ   R2,#15
  880 ADRNE R0,wrong_mode_error
  890 SWINE "OS_GenerateError"
  900 MOV   R10,#640
  910 MOV   R4,R4,LSR #1
  920 :
  930 .check_zoom_number
  940 CMP   R3,#6
  950 ADRHI R0,bad_zoom_number_error
  960 SWIHI "OS_GenerateError"
  970 :
  980 .check_bank_numbers
  990 TEQ   R6,R7
 1000 ADREQ R0,same_bank_error
 1010 SWIEQ "OS_GenerateError"
 1020 :
 1030 .get_bank_addresses
 1040 MOV R0,#112:MOV R1,R6:SWI "OS_Byte
":MOV R9,R1
 1050 ADR R0,list:ADR R1,scr_addr:SWI "O
S_ReadVduVariables"
 1060 LDR   R12,scr_addr
 1070 MOV R0,#112:MOV R1,R7:SWI "OS_Byte
"
 1080 ADR R0,list:ADR R1,scr_addr:SWI "O
S_ReadVduVariables"
 1090 MOV R0,#112:MOV R1,R9:SWI "OS_Byte
"
 1100 LDR   R11,scr_addr
 1110 :
 1120 .check_bank_addresses
 1130 TEQ   R11,R12
 1140 ADREQ R0,insufficient_mem_error
 1150 SWIEQ "OS_GenerateError"
 1160 :
 1170 .zoom
 1180 TEQ   R3,#0:BEQ   zoom_zero
 1190 :
 1200 ADD   R12,R12,R4:MUL   R5,R10,R5
 1210 ADD   R12,R12,R5
 1220 :
 1230 MOV   R8,R10,LSR R3 ;R8=block widt
h
 1240 MOV   R0,#256
 1250 MOV   R9,R0,LSR R3  ;R9=block heig
ht
 1260 SUB   R7,R10,R8     ;R7=inc needed
 1270 MOV   R1,#1
 1280 MOV   R6,R1,LSL R3
 1290 RSB   R5,R10,R10,LSL R3
 1300 :
 1310 TEQ   R3,#1:BEQ   zoom_one
 1320 :
 1330 .zoom_two_to_six
 1340 MOV   R4,R8
 1350 .next_pixel
 1360 LDRB  R0,[R12],#1
 1370 ORR   R0,R0,R0,LSL #8
 1380 ORR   R0,R0,R0,LSL #16
 1390 MOV   R2,R6
 1400 .next_column
 1410 MOV   R3,R11:MOV   R1,R6
 1420 .next_block
 1430 STR   R0,[R3],R10
 1440 STR   R0,[R3],R10
 1450 STR   R0,[R3],R10
 1460 STR   R0,[R3],R10
 1470 SUBS  R1,R1,#4
 1480 BNE   next_block
 1490 ADD   R11,R11,#4
 1500 SUBS  R2,R2,#4
 1510 BNE   next_column
 1520 SUBS  R4,R4,#1
 1530 BNE   next_pixel
 1540 :
 1550 ADD   R12,R12,R7
 1560 ADD   R11,R11,R5
 1570 SUBS  R9,R9,#1
 1580 BNE   zoom_two_to_six
 1590 LDMFD (sp)!,{PC}
 1600 :
 1610 .zoom_zero
 1620 MOV   R10,R10,LSL #8
 1630 .next_block2
 1640 LDMIA R12!,{R0-R9}
 1650 STMIA R11!,{R0-R9}
 1660 SUBS  R10,R10,#40
 1670 BNE   next_block2
 1680 LDMFD (sp)!,{PC}
 1690 :
 1700 .zoom_one
 1710 MOV   R4,R8
 1720 .next_pixel_pair
 1730 LDRB  R0,[R12],#1
 1740 LDRB  R1,[R12],#1
 1750 ORR   R0,R0,R0,LSL #8
 1760 ORR   R1,R1,R1,LSL #8
 1770 ORR   R0,R0,R1,LSL #16
 1780 STR   R0,[R11,R10]
 1790 STR   R0,[R11],#4
 1800 SUBS  R4,R4,#2
 1810 BNE   next_pixel_pair
 1820 ADD   R12,R12,R7
 1830 ADD   R11,R11,R10
 1840 SUBS  R9,R9,#1
 1850 BNE   zoom_one
 1860 LDMFD (sp)!,{PC}
 1870 :
 1880 .list EQUD 148:EQUD -1
 1890 .scr_addr EQUD 0
 1900 ]:NEXT
 1910 :
 1920 SYS "OS_File",10,"Zoomer256",&FFA,
,code%,O%
 1930 SYS "OS_Module",11,code%,P%
 1940 *MODULES
 1950 *HELP Zoom
```

USING THE DEMO

The best way to illustrate the use of Zoomer is to look at the demo. Type in listing 2, and save it away. Make sure that you have a mode 13 screen called ZoomScr, in the current directory of your disc, and run the program. The Zoomer module, and the ZoomScr screen will now be loaded in, and each time that you click *select* on the mouse, the image will double in size. The smallest movements of the mouse will now cause the whole image to pan across the screen without a glitch, and at the utmost speed. To reduce the magnification, just click on *adjust*.

A look at the program will show what is happening. First *FX114 sets up dual screens, then the mode 13 demo screen is loaded into screen RAM (the non-shadow bank).

The REPEAT loop then checks the mouse, and sets the X and Y co-ordinates for the zoom,

Continued on page 23

4-SLOT AND 2-SLOT BACKPLANES FOR ARCHIMEDES 305 AND 310

Before you can fit any podules to a 300 series Archimedes, a backplane is required. Our new four-slot backplane may be fitted to any 305 or 310 Archimedes computer, and accommodates up to four single width podules. Installation is very straightforward, and no soldering is needed.

- ★ **Accepts up to 4 podules**
- ★ **Ideal for use with the Computer Concepts RAM/ROM Podule**
- ★ **Money-back guarantee**
- ★ **Available as an upgrade**

UPGRADES

If you already have a 2-slot backplane, then our 4-slot version may be obtained at a discount price. This offer is for a limited period only - please telephone for details.

Prices (excluding/including VAT)

4-way backplane (incl. fan)	£58.00	£66.70
4-way backplane (no fan)	£48.00	£55.20
2-way backplane (incl. fan)	£33.00	£37.95
Computer Concepts ROM Podule	£46.95	£54.00
As above with battery back-up	£55.65	£64.00
4-way backplane as upgrade	£31.50	£36.22
62256 32K RAM chips	£11.00	£12.65
Postage £1.00		

HOW TO ORDER: Cheques (payable to "IFEL"), POs or official orders welcome. Access orders accepted on (0752) 847286. Please allow £1.00 P & P.

IFEL, 36 UPLAND DRIVE, DERRIFORD, PLYMOUTH PL6 6BD (0752) 847286

"FIRST"

ADVANCED STATISTICAL SYSTEM

For Acorn Archimedes; BBC Models B, B+ and Master; "Industry Standard" machines

A powerful and practical tool for Research, Industry, Business, and Teaching

Integrated, interactive, robust, fast, accurate, modular. Designed to facilitate data critique.
Many data entry options. Full data management and formatted display. Extensive data transforms.
Handles missing variables and data subsets. Scatterplots, regression plots, function plots.
Univariate statistics. Paired and unpaired t tests. Chisquare tests. Nonparametric methods.
Unrivalled REGRESSION facilities eg weighted, through the origin, standardised. Full statistical information.
Residuals, fitted and predicted values. Much more. Correlations. ANOVA. Contour plots of regressions.
Regression DIAGNOSTICS - VIFs, influential points etc. Automatic warnings.
ROBUST regression - many influence functions. Powerful NONLINEAR least squares
Time Series, eg moving averages, exponential smoothing, causal models.
One- Two- and Three-way ANOVA subsystems. Distributions generator. Histograms. Linear Calibration.
Cluster analysis. Many other powerful analytical and descriptive features. Full utilities.

Price £120-£180 (machine and version dependent). Special RISC USER discount available.

There's far too much to describe here. Get full information now from:

Serious Statistical Software, Lynwood, Benty Heath Lane, Willaston, South Wirral L64 1SD Tel. 051 327 4268

THE RISC OS DESKTOP

by David Spencer

We take a look at the enhanced Desktop environment offered by RISC OS.

The Desktop Manager in RISC OS, while initially appearing similar to that in Arthur 1.20, is in fact totally different in operation. Rather than existing as a stand-alone program, the RISC OS Desktop Manager is a relocatable module which provides a 'launch pad' for genuine applications (such as word processors, drawing packages etc.). This means that you are no longer restricted in what can be run from the Desktop. *Any* application can be installed using the Desktop and run in a multi-tasking environment (see separate article in this issue). We will come back to this later.

STARTING THE DESKTOP

The RISC OS Desktop is started in the same way as for Arthur 1.20. That is, either by configuring it as the default language, or issuing *DESKTOP. One very welcome addition is the fact that the Desktop will quit immediately if it is started as the default language and there is an EXEC file open. This means that discs with a boot option set using *OPT4,3 can now be booted properly with Shift-Break from within the Desktop. The *DESKTOP command actually closes any EXEC files, so *DESKTOP cannot appear in an autoboot sequence except as the last command.

APPEARANCE

The initial appearance of the RISC OS Desktop will be familiar - a large blank background with an icon bar across the bottom. The colours are different to those under Arthur, with subtle shades of grey being adopted. Whereas before the icon bar could not be overwritten, RISC OS allows other windows to move over it.

The contents of the icon bar are also much the same as before. Icons relating to filing system devices appear to the left, and icons representing utilities and applications are at the right.

The filing system icons that are present depend on the configuration of the machine. There are a number of floppy and hard drives, and a network symbol if Econet is present. There is also a 'chip' icon to represent the RAM filing system if this is enabled.

Initially, the number of icons on the right-hand side of the bar is less than that on the Arthur Desktop. Gone is everything except for the palette icon representing the Palette Manager, and this is now of a different design. The only other icon is an Archimedes 'A' which represents the *Task Manager*. The purpose of this is described in the article on multi-tasking elsewhere in this issue.

FILES AND THINGS

As before, clicking on one of the file device icons brings up a catalogue for that device. In the case of Econet, if you are not currently logged on you are prompted for a username and password. Any window displaying the contents of a directory is called a *Directory Viewer*, and has as its title the full pathname of the directory it is displaying.

The format of the file display can be selected to be one with large icons and filenames, or a more compact form using small icons and filenames, or a full format that includes all file information along with small icons. The choice of icon is determined by the filetype of each file, and is fairly logical. The icons for system type files, such as relocatable modules, are based around the Archimedes 'A', while text files show a pen and paper, and Basic files a program listing. Individual applications can set up their own icons for particular file types that they recognise. For example, a spreadsheet might define a certain filetype to be reserved for its own data files, and then define a spreadsheet-like icon to represent this.

Clicking on any entry in a Directory Viewer will select that entry and highlight it by reversing its icon colours. More than one object can be selected at once by using the adjust button to perform the selections. Once an object has been selected you can perform operations such as renaming it or deleting it, simply by pressing the Menu button and selecting the appropriate option. Double clicking on a file in a Directory Viewer will cause that file to be run. The exact effect of this depends on the filetype of the file. For example, a Basic program will be run directly, while a text file might be loaded into a text editor.

Sub-directories are shown within a Directory Viewer as a folder icon. Double clicking on a folder will cause the sub-directory to be opened and its contents displayed in a new Directory Viewer.

RISC OS supports a special type of directory known as an *Application*. As its name suggests, such a directory contains the individual components that make up an application. For example, a Desktop Publishing application might consist of the program itself along with a help file and a set of standard page layouts. These individual parts, known as *Resources*, would all be put together in a single application directory. When an application is double-clicked on, rather than opening the directory, the application is started up using those resources which RISC OS expects to find in that directory. Holding the Shift key down when double-clicking on an application does in fact allow you to open it as a normal directory in order to access the individual resource files. Another feature of an application is that it can define the icon used to display its directory in the Viewer. This will usually be similar to the icon used for its data files so that the two are easily associated.

A useful feature of Directory Viewers is that files can be copied simply by dragging their icons from one Viewer to another. If more than one file is selected then they are all copied in one go. This method will even copy files between filing systems. There is no move option, so this must be achieved either by using rename, or by copying the file and deleting the original.

APPLICATIONS

As mentioned earlier, the Desktop Manager itself contains no applications or utilities. Even the *Palette Manager* and the *Task Manager* are separate applications contained elsewhere in the RISC OS ROMs.

Other applications will be stored on disc in application directories, as described above, and have to be installed before they can be used. This can be done by double-clicking on the application's icon, or by double-clicking on a data file belonging to the application. In the latter case the application is installed, and the data file loaded automatically.

Each application, once running, will have one or more windows open on the screen. Because of the multi-tasking environment, all the applications installed will remain active at all times. However, the user can only respond to one application at a time, being determined by the window over which the mouse pointer is positioned, and the window that contains the text entry caret. An exception to this is the so-called 'hot keys' system which allows any application to recognise a particular key press whatever the situation is. For example, pressing Alt-H could be made to pop up a help box at any time.

There are essentially two forms of applications. Transient utilities start running when they are installed and are automatically removed when all their windows are closed (for example, the CPU Usage utility). The other type of application places an icon on the icon bar when it is installed (for example, ArcEdit). It can then be started by clicking on this icon or dragging a file onto the icon. The application is stopped by closing its windows, and restarted by clicking on the icon again.

THE PALETTE MANAGER

The *Palette Manager* allows the colour palette to be redefined, saved and loaded, in much the same way as the Arthur Desktop. As before, colours are changed by selecting the appropriate colour and dragging sliders to adjust the Red, Green and Blue components.

An important new feature of the RISC OS window system is the ability to run in any screen mode, including 256 colour and 132 column modes. The palette utility allows the user to select which mode is used, and a default can also be set up using *CONFIGURE.

The article on RISC OS multi tasking, elsewhere in this issue, explains application programs in more detail. RU

RISC OS

MULTI-TASKING WITH RISC OS

by David Spencer
Advance information on the most exciting feature in RISC OS.

This is the first of two articles in which we will be looking at the multi-tasking system offered by RISC OS. We will concentrate here on a user's view of the system.

RISC OS multi-tasking relies totally on the WIMP Manager, and is therefore closely related to the desktop environment. If you have not already read the article in this issue about the RISC OS Desktop Manager, then it might be useful to read it now.

WHAT IS MULTI-TASKING?

As the phrase suggests, multi-tasking is the ability to execute two or more programs simultaneously. Obviously, executing more than one program at the same time (in a truly simultaneous way) is not possible on a single processor system. Instead, you would need a separate processor for each program which was running.

Therefore, the best that can be achieved on a single processor system is to allow several programs to run concurrently over a period of time, though at any given instant only one program is actually executing. This is accomplished by switching between each program in turn at a rate sufficiently high to give the effect of multi-tasking. An analogy to this is a television picture. The actual picture is made up of a series of frames. However, because these are displayed at a high rate (fifty a second), the overall effect is of a continuously moving picture.

COMMUNICATING WITH THE USER

One question that should immediately spring to mind is how the output from all the programs converges onto a single screen, and how the user decides which program he is communicating with. This problem is very conveniently solved using a window environment. Each program can display its output in one or more windows. Because there can be many windows on the screen at any time, the output of several programs can be shown simultaneously. The user can decide the positioning, and size etc., of these windows in the same way as with any WIMP based program on Arthur.

Input is handled in much the same way. In the case of the mouse, the user can simply move the pointer to the correct place and click the mouse buttons as necessary. Which program deals with the button clicks is determined by which window the pointer is over. For example, clicking *menu* over a window owned by a clock utility might bring up a menu allowing the time to be changed, while clicking *menu* over a word processor's menu could bring up a menu of editing options.

The particular program that receives any key presses is controlled using a concept known as *input focus*. Of all the windows open, one and only one, has the input focus at a given time. This particular window is normally displayed with a different colour border. Any key presses are passed to the program whose window currently has the input focus. Typically, a window is given the input focus by clicking *select* within that window. If this isn't clear, try it out with the Note Book and a Diary page on the Arthur Desktop.

APPLICATIONS

Up until now we have blindly talked about a number of programs running together, but these can't be any old programs - they must be of a special form. From the user's point of view this will generally mean that they must be programs specially written to multi-task under RISC OS. In particular, as has already been said, all multi-tasking programs must run under the window system. However, not all window based programs are multi-tasking. A window-based program written for use with Arthur will run under RISC OS, but all the other programs will be suspended for the entire time that the program is active.

Subject to the above provisos, there is really not much restriction on what tasks a multi-tasking application can perform. However, most applications will fall into one of two groups. The first of these are the so-called *utilities*. These consist of 'handy' programs such as a clock or calculator. The second type of applications are referred to by Acorn as *editors*. An *editor* in this context means a program that reads a data file, allows the user to modify it as necessary, and then saves it again. An obvious example of an *editor* is a word processor. Another example is a database manager.

INSTALLING APPLICATIONS

In most cases, all control of multi-tasking applications will be performed from within the Desktop. Whenever the Desktop is started, a number of resident applications will be installed ready for use. One of these is the application that controls the handling of files from within the Desktop. The other two standard ones are the *Palette Manager*, which is described in the Desktop article in this issue, and the *Task Manager* which is dealt with below. These applications are all written in ARM machine code and form part of RISC OS itself. All other applications will be stored on disc (or on a network), and are installed by double-clicking on their icon in a directory viewer.

Once installed, applications can appear in one of two ways. Some applications, particularly utilities, will open a window immediately and start running in that window. If the user closes the window by clicking on the 'Quit' box then the application is removed from memory. An example of this is the CPU Usage utility on the RISC OS Welcome Disc. With the second method, typically used by *editors*, the application does not open any windows, but instead places an icon on the right-hand side of the icon bar. Examples of this include Arcedit and the resident applications (the Palette and Task Managers). In this case, clicking on the icon opens the application's windows, and even if they are subsequently all closed by the user, they can be opened again by clicking on the icon again. Clearly this method is aimed at 'long term' applications which can be called up whenever needed.

It is also possible to start applications via their data files. You can double-click on a datafile of a type recognised by a particular application, and this will install and start that application. Alternatively, if the application is already installed, you can drag the file's icon and drop it either over an open window of the desired application, or over the icon for that application on the icon bar.

All applications offer a menu option to quit from them, and this automatically removes them from memory. Additionally, the *Task Manager* (see below) offers an 'Exit' option which removes all applications and quits the Desktop.

THE TASK MANAGER

This is a resident task whose job is to give the user control over other tasks. We have already mentioned one of its options, namely quitting the Desktop, and another allows the user to enter star commands. However, the *Task Manager*'s major role is in the area of memory management. Obviously, all applications that are installed have to share the available memory. The *Task Manager* can display a list of all 'memory users', along with the amount of memory that they use. This list includes not only all the current applications, but also things such as screen memory, module memory, system sprite area, etc. Any memory not currently allocated is also shown.

Each entry in the *Task Manager*'s display also has a slider, the length of which is related to the amount of memory allocated. It is possible to drag these sliders to change the amount of memory allocated. For example, the Screensize could be reduced to allow more application workspace simply by dragging the appropriate slider backwards. There is no need for any *CONFIGUREs or hard resets.

Next month we will take a look at multi-tasking from the programmer's point of view. RU

Putting the Pro into Artisan

Roger Burg examines Clares' latest graphics package, ProArtisan.

Clares' Artisan was one of the very first software releases to be produced for the Archimedes, and has established itself as probably the best of the art packages so far. Now Clares has released ProArtisan, not a replacement, but a more professional and comprehensive approach to the same task, using the 256 colours of mode 15 in contrast to the 16 colours of artisan.

ProArtisan is supplied on two discs together with a handsomely produced manual of 113 pages, including full colour reproductions of some of the example screens supplied with the package. Once loaded, ProArtisan will be immediately familiar to anyone who has already used Artisan (see review in RISC User Volume 1 Issue 1).

It is an extremely easy-to-use mode 15 paint program. Routines accompany it, on disc, to convert from mode 12 to mode 15, but no other mode is supported. It does not attempt any perspective features, and although the absence of such facilities is common to many graphics packages, it still appears as an important restriction in most representational work.

ProArtisan's text facilities are limited. It's aimed at screen images only. Printer dumps are just representations of the screen, and there is no facility to design images of greater resolution or size than the standard mode 15 display. If you want something to design sprites for another mode, or produce high resolution printed output, you want something else.

That said, ProArtisan is a lovely package to use. There are stacks of features in it to cope with most requirements. ProArtisan has removed some of the restrictions of Artisan and now uses a 256 colour mode. The mode 15 palette and resolution adopted allow textures the like of which I have not seen on screen before.

All the functions (well most of them) are so fast that one simply forgets about response times. This transforms features which might otherwise be of limited usefulness. The trick of grabbing an area of the screen, with or without background, to place it elsewhere, has become an indispensable tool to me now. It allows me hundreds of minor adjustments which on other software would have been too slow or too awkward to make.

FEATURES

I can't really do justice to every single aspect of ProArtisan in this review, so I've summarised the main menu options, and then described some features in greater detail to give a representative idea of the whole.

The select (left-hand) mouse button usually activates the current option. The menu (middle) button calls up the colour palette menu (one click) or current menu (double click). The adjust (right-hand) button is used either to *undo* your last bit of painting (everything since the last menu was displayed), or to select alternative actions from some menus. The keyboard is hardly used at all.

COLOUR SELECTION

The 256 colours have been arranged in groups of shades of a particular colour, which looks much more attractive and sensible than the more often used colour number order. As well as selecting a colour from the palette, a colour can also be picked up from anywhere on the screen (and the same is true in zoom mode).

THE DRAW MENU

The drawing options include nib size and four nib shapes as before, a redefinable nib, a nice airbrush, magnify (x2, x4 and x8), flood fills which may be variously graduated, re-colour, and wash. The latter smudges a part of the image by averaging adjacent colours. It can look watery, but is more accurately like smudging chalks, with the advantage that the screen can be un-smudged. I was very sceptical about this at first, but it persuaded me. It's not 'very useful', but 'useful' it certainly is.

The zoom facility allows the enlarged picture to be scrolled within a window, but apart from selecting a colour from the palette, only a simple pixel filling option is possible. It would be advantageous to use larger brush sizes and other facilities when working in this mode.

THE BANDED MENU

The rubber-banded shapes menu includes straight lines, many polygons and curved shapes, options to operate and adjust a gridlock, and the facility to design and draw a smoothing (Bezier) curve. This last seems to me to be implemented particularly awkwardly, quite out of character with the rest of the package, although smoothing curves have seldom been included on Acorn software, and it's good to see this omission corrected. Oddly the rubber-banded shapes all ignore the current nib size.

SPRITE MENU

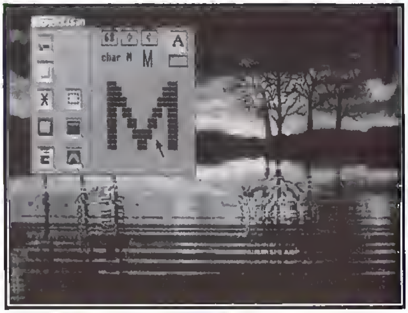
In ProArtisan, several significant new functions have been added. There are functions to create and remove masks and to stretch, though not rotate, sprites so that the earlier cut and paste options are almost redundant. Ingenious options have been added to define sprites on the zoom screen or by an irregular hand-drawn outline. There is no option to simply 'colour in' the pixels to be included in a masked sprite, and I suspect this is the solution towards which Clares is groping.

THE TOOL MENU

The Tool menu contains the 'left-overs' and a bit more. You can clear the screen, use 26 different disc-based fonts (not Acorn's fancy fonts), call your printer dump routine, reduce a selected part of the screen to a monochrome line drawing before printing, adjust the mouse response and use star commands.

ProArtisan provides 19 different printer dumps covering some 10 printer types, including Integrex and HP colour printers, Brother, Panasonic and other 24-pin printers, Epson FX, RX and MX 9-pin printers and compatibles. Any other printer dump, provided it is in the form of a module, can be easily installed and used. Really, a 24-pin monochrome printer, or a colour printer, is needed to do justice to pictures produced with ProArtisan.

The fonts in ProArtisan seem to be an improvement over those in Artisan. There, spacing was so wide that I adjusted every line by hand, letter by letter. Oddly there is no facility for anti-aliasing nor for scaling. It should at least be possible to adjust the size and proportions of text in a package of this quality.



Using the font designer from the Tool menu

There is also a *global toner* feature, which converts any pre-defined rectangular area to a monochromatic range of tints based on a single colour. The Tool menu also allows the original Artisan *magic brush* feature to be applied instantaneously to any area of the screen, avoiding the need to paint with individual brush strokes over the same area. The magic brush allows defined colour changes to take place without affecting other colours.

The Tool menu also provides the means whereby any selected sprite may be distorted over almost any area giving in many instances a three dimensional, though not truly perspective effect.

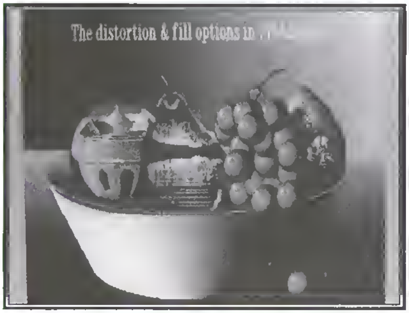
FILING MENU

As well as saving and loading screens and sprites, disc filing options now include the saving and loading of Bezier curves, magic brush sets, fonts and pattern fills, all most welcome, and a compacted screen save, which is now the default.

SOME GENERAL COMMENTS

The graduated fill routine is particularly nice. It allows you to fill an area with a range of tints or tones. So, if you draw a rectangle, you can 'fill' it light on one side, through to dark on the other, so that it appears as a cylinder.

This extends to widening the steps of shading so that you can fill bigger cylinders or narrow ones correctly. It will also fill in vertical bands as for a horizontal pipe, and in concentric shades, like the shading of a sphere. The bands of shading can be hard-edged like a faceted pillar or faded irregularly, like an airbrush effect. And the routine is impressively fast.



Distort and fill options

All this was very hard work in Artisan, and greatly restricted the already limited range of colours, but in Artisan, you could redefine the palette to get the blends you wanted (from the Archimedes total colour range of 4096). In ProArtisan the palette consists of a fixed 256 colours (the default palette definitions), so although you can modify the range of colours which it has automatically selected for a graduated fill, those available seldom form the sequence which you really want.

THE MANUAL

The manual deserves comment. In spite of its 113 pages, it seems short. It is well set out, illustrates the appropriate icons and menus well, and is exceedingly clear. There is a quick guide, most useful for those already familiar with Artisan and just raring to go, and a comprehensive tutorial guide, with suggested exercises for each menu.

CONCLUSIONS

The minimal use of the keyboard in my view is sad. ProArtisan is a slave to the mouse, and to its limitations. In particular, the 'undo' should not normally be on the mouse buttons, and it is still not possible to pick points beyond the edges of the screen, simply because the mouse cannot reach them.

I'd also prefer fewer colours but with well-programmed mixes, and much higher resolution, though this is to some extent the restriction of the screen modes (i.e. Acorn's problem not Clares'). The restriction to a single screen mode is a big practical limitation.

There is a need for precise cursor-control, preferably with the cursor keys. The gridlock facility (of 'snapping' to grid lines) is not what is required, and in any case is not sufficiently adjustable.

In all honesty though, first Artisan, and now ProArtisan, have achieved so much for the Archimedes that it seems to be churlish to be critical at all. But there is still plenty of scope for the further development of good graphics software for the Archimedes.

And finally, is it worth the near £170 asking price? This certainly seems steep, particularly when the Archimedes is still striving to establish itself. And even if ProArtisan is aimed more at the professional user that hardly warrants a price which will undoubtedly make many potential purchasers think twice.

ProArtisan is the best art package currently available for the Archimedes. If you need its capabilities it will do an excellent job. My advice is to try it out if possible before purchase. If you are still impressed, then go ahead. Certainly the sample pictures supplied with the package are absolutely stunning.

Product	ProArtisan
Supplier	Clares Micro Supplies 98 Middlewich Road, Rudheath, Northwich, Cheshire CW9 7DA. Tel. (0606) 48511
Price	£169.95 inc. VAT.

Note: first orders will be supplied as a limited edition in a polished wood case.

RU

EXPANSION CARDS FOR THE ACORN ARCHIMEDES COMPUTER SYSTEM

IEEE488 INTERFACE a full implementation of the standard for automatic test and measurement systems

16 BIT PARALLEL I/O two 16 bit input or output ports with handshake lines for digital control applications

DUAL RS423 SERIAL INTERFACE for communicating with two additional RS423 or RS232 devices eg printers, plotters, instruments, etc

12 BIT ANALOGUE I/O in development

All the above high performance expansion cards are supplied with high level software for ease of use and a comprehensive user guide.
Take advantage of Intelligent Interfaces' expertise and purchase a complete Archimedes Computer System.

Officially appointed Acorn Scientific Dealer.

I I

Intelligent Interfaces Ltd
43b Wood Street
Stratford-upon-Avon
Warwickshire
CV37 6JQ
Tel: 0789 415875
Telex: 312242 MIDTLX G

Arthur 1.2

Getting to Know Arthur (Part 2)

Mike Williams continues to provide the low down on talking to Arthur, the Archimedes operating system.

Last month I described how the *SET command could be used to set up so-called aliases. An alias is a user-selected name that once initialised will be recognised by the operating system (OS) as equivalent to some predefined string. For example:

```
    *SET Alias$SCREEN CONFIGURE Screensize20
```

would make the command:

```
    *SCREEN
```

exactly the same as:

```
    *CONFIGURE Screensize 20
```

You could establish a number of aliases with names such as SCREEN20, SCREEN40 etc to provide short star commands for selecting various memory allocations for screen displays.

However, we can still use a single alias command to set more than one screensize by using a parameter. Such an alias command could be set up using:

```
    *SET Alias$SCREEN CONFIGURE Screensize %0
```

The %0 indicates a parameter whose value will be supplied when the alias is called.

```
    *SCREEN 20
```

would set a screen memory size of 20 pages, while:

```
    *SCREEN 40
```

would reserve 40 pages for this purpose. The obvious disadvantage of all this is that the Ctrl-Break needed to enable the new configuration will also destroy your current aliases!

Simple aliases like this can be quite useful, as they are quicker and easier to type in than the original command, and maybe easier to remember. Indeed, we can take the use of parameters and aliases much further, but one more example will suffice here. Up to 10 separate parameters may be defined, and these are represented as %0 to %9. Each parameter can appear more than once if necessary. In addition, a single alias may be used as a substitute for more than one command.

Suppose we want to create an alias called INIT which will allow us to set both screensize and spritesize. This is simply achieved by writing:

```
    *SET Alias$INIT CONFIGURE Screensize
    %0|MCONFIGURE Spritesize %1
```

If we subsequently type:

```
    *INIT 20 10
```

this would configure screensize to 20 and spritesize to 10, equivalent to entering the two separate commands:

```
    *CONFIGURE Screensize 20
    *CONFIGURE Spritesize 10
```

When an alias, as INIT in this example, is called, the first parameter is assigned to %0, the second (separated by a space) to %1 and so on. Notice also the '|M' which represents Ctrl-M or Return in the alias definition, just as it does in function key definitions.

One final point on alias parameters is worth mentioning. If a command is to take several parameters, then rather than specifying them all individually with %0, %1 etc., the notation %*0 will grab all the parameters supplied.

Any suitable string assigned to an alias may include further commands. Thus:

```
    *SET Alias$PAGE ECHO |<14>
    *SET Alias$SCROLL ECHO |<15>
```

would allow *PAGE to set paged mode (Ctrl-N or ASCII 14) and *SCROLL to set scroll mode (Ctrl-O or ASCII 15). ECHO used in an alias definition can be very powerful - see hint in Volume 1 Issue 5.

One further command to note here is *UNSET. This may be used to delete any operating system variable (except any permanent system variables), set up with the *SET command. It also accepts wildcard characters, so:

```
    *UNSET Alias$*
```

will delete all current aliases.

OTHER *SET COMMANDS

There are two further star commands similar to the basic *SET. The command *SETMACRO can be used to assign an *expression* to an OS variable which is evaluated each time the variable is referenced. One example might be:

```
    *SETMACRO CLI$PROMPT <SYS$TIME>
```

which sets the OS prompt (see last month) to be the current time (the time when the prompt occurs). The command:

```
*SET CLI$PROMPT <SYS$TIME>
```

on the other hand, would set the prompt to be the time when the *SET command was executed (i.e. fixed).

The other command, *SETEVAL is used to assign a *value* to an OS variable, by evaluating an expression, instead of a string as with *SET. The use of SETEVAL in this way is similar to the use of the EVAL function in Basic. Thus:

```
*SETEVAL fred 0
```

would initialise the OS variable *fred* to 0, and subsequently:

```
*SETEVAL fred fred+1
```

would increment the value of *fred*. Typing or executing:

```
*SHOW fred
```

would display the current value of *fred*.

Do remember that the OS variables we are talking about are quite separate from those used in Basic, even though they may appear very similar. These variables are used when communicating directly with Arthur (which gives the "*" prompt).

BASIC EQUIVALENT COMMANDS

A number of operating system commands have equivalents in Basic. Whichever form is used, it is the same in-built routine in the operating system which is being accessed, but it does mean that these functions are accessible whether we are in Basic or Arthur. Thus *EVAL used with OS variables performs a similar task to that which the EVAL function does with Basic variables; it evaluates a given expression.

Then there is *ERROR. This command can be used to create error numbers and corresponding messages of your own choosing (just like ERROR in Basic). These can be used in your Basic programs, and will be dealt with by any normal error-handling routine just as with other errors.

For example, suppose you were to write, in a Basic program:

```
IF x<0 and x>100 THEN *ERROR 101 Value
out of range
```

If this was executed with the value of x either less than zero or greater than 100, then an error would be generated as defined by *ERROR. In an error-handling routine, the variable ERR would give the value 101, and REPORT (or REPORT$) would give the text message "Value out of range".

The *IF command provides an IF-THEN or IF-THEN-ELSE syntax, again with the same implications as in Basic. For example:

```
*IF Sys$Time<"12" THEN ECHO Good
Morning ELSE ECHO Good Afternoon
```

will display an appropriate message depending on the time of day (note the quotes round the '12' because Sys$Time is a string variable - see last month).

Using commands like this enables complete programs to be written which will be directly executed by Arthur, not by Basic. Again this is all good material for an EXEC file.

Finally, in this section, *TIME may seem as though it will be the same as Basic's TIME, but is in fact more like the Basic pseudo-variable TIME$. It reproduces the current date and time in a standard format. In Basic, TIME$ is a string which can be printed or otherwise manipulated, while *TIME displays the complete date and time on the screen.

BBC MICRO COMMANDS

Some of the commands recognised by Arthur are simply there to provide compatibility with earlier BBC micros, though that is not to say that they do not provide a useful function on the Archimedes. *KEY can be used to program the function keys, and also the Print, Copy and the four cursor keys. The use of *KEY is covered adequately in the User Guide, but remember the use of codes like |M or |N to represent control codes like Return or Ctrl-N.

The *SHADOW command is mainly for compatibility with the Master 128 and Compact. If no parameter (or a value of 2) is provided, bank 2 of screen memory will be used; if the command is followed by a '1' then memory bank 1 will be used (see our recent series on *Animating Archie* for more information on bank switching).

Lastly, *TV can be used to adjust the vertical position of the display on your monitor, and to control the selection of interlace, as described in the User Guide.

MISCELLANEOUS COMMANDS

There are four commands which I want to deal with here. The *IGNORE command can be used to set the 'printer ignore' character. For most printers this is set to zero (the null character). If your printer executes a double line feed at the end of each line, try setting the printer ignore character to 10 (*IGNORE 10).

The next two commands are very similar and best treated together, though quite different in application. They are *GO and *GOS. The former may be used to call any machine code routine by specifying its start address. It is Arthur's equivalent to Basic's CALL, except that after *GO a return is not usually made back to Arthur. Unless you are into machine code programming this call will have little value for you (machine-code programmers should refer to the *Programmer's Reference Manual)*. The other command, *GOS (short for *GO Supervisor), simply invokes Arthur, and in that respect is very similar to QUIT in Basic.

The last command in this group is *HELP which can be used to display information on other star commands, including those supported by filing systems and other modules. Simply type:

```
*HELP
```

for more information on this command.

FX CALLS

The last OS command to mention gives access to many different functions. *FX is already well known to previous owners of BBC micros of all kinds, but performs fewer functions in its Archimedian form. In essence, *FX provides a convenient way for all users to access a range of operating system routines (known as Os-Byte calls). In each case *FX is followed by a number which specifies the particular function to be carried out, and depending upon the call used, by one or two parameters. The FX calls are listed in the User Guide, so I do not propose to cover them here.

I hope this discussion will have proved useful to many readers, and given some insight into the Archimedes operating system, Arthur, and the commands which it recognises. More information is contained in the *Programmer's Reference Manual.* Of course, the Archimedes responds to many more star commands than those that I have described. These are the ones used by filing systems and any modules in your system. But that is another story. RU

ARC FULL-SCREEN ZOOM AND PAN (continued from page 11)

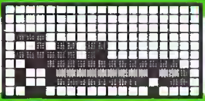
and the magnification number. Then *Zoom is called (using OSCLI), and *FX 113,2 is used to ensure that screen two (the shadow screen) is the one which is displayed. Because no bank numbers are supplied with the *Zoom call, the defaults of 1 and 2 are used. This means that screen 1, into which the screen image was originally loaded, is always the source, and the shadow bank, which is the one on display, is always the destination.

The code is so fast that there is no need for the use of alternate bank switching to avoid flicker.

Listing 2

```
   10 REM >ZoomDemo
   20 REM by Mike Ironmonger
   30 :
   40 ON ERROR OSCLI("FX 114,1"):MODE0:R
EPORT:PRINT" at line ";ERL:END
   50 *Zoomer256
   60 *FX 114
   70 MODE 13:OFF:Z%=0:*Pointer
   80 *ScreenLoad ZoomScr
   90 :
  100 REPEAT
  110 MOUSE MX%,MY%,B%:MY%=1024-MY%
  120 IF B%=0 TIME=26
  130 IF TIME>25 AND B%=4 AND Z%<6 Z%+=1
:TIME=0
  140 IF TIME>25 AND B%=1 AND Z%>0 Z%-=1
:TIME=0
  150 X%=MX%-(1280>>Z%)/2
  160 Y%=MY%-(1024>>Z%)/2
  170 IF X%<0 X%=0
  180 IF Y%<0 Y%=0
  190 WAIT:OSCLI ("ZOOM "+STR$Z%+CHR$32+
STR$X%+CHR$32+STR$Y%)
  200 *FX 113,2
  210 UNTIL FALSE
```

RU



USER SPRITES

by David Spencer

Prepare the way for RISC OS multi-tasking by moving over to user sprites. They give more flexibility and greater speed.

Most simple programs that use sprites store them in the system sprite area set up using the command *CONFIGURE SpriteSize, and manipulate them using commands such as *SLOAD and *SCHOOSE, and plot them using PLOT &ED or the like. (See part one of *Animating Archie* in RISC User Volume 1 Issue 8). This method has the advantage that it is straightforward, and is also fully compatible with the Graphics Extension ROM (GXR) on earlier Acorn machines. However, there are a number of disadvantages of using this technique. Firstly, if there is insufficient sprite RAM allocated, then *CONFIGURE must be used to allocate some more, and the computer hard reset. Secondly, and of far more importance, multi-tasking programs running under RISC OS might compete to use the system sprite area. For example, one program might load in a set of new sprites totally overwriting all the sprites used by other programs.

What is needed is a method by which each program can have its own 'personal' sprite area. This would resolve any problems of programs interfering with each other. Additionally, if each program could choose the size of its sprite area, then the problem of not having enough space is also resolved. The Archimedes provides so-called *User Sprites* for just this purpose, and it is these which are described here.

The system of User Sprites is in fact much more powerful, and faster, than using star commands and PLOT, but in this article I shall just concentrate on the common sprite operations. These are clearing the sprite area (*SNEW), deleting a sprite (*SDELETE), loading sprites (*SLOAD), saving sprites (*SSAVE), plotting a sprite, and grabbing a sprite from the screen. Details of all the other possible operations can be found in the chapter about sprites in the *Programmer's Reference Manual*.

The first step in using User Sprites is to allocate a block of memory in which to store the sprites. This is where the term 'User' comes in, because it is up to the user's program to reserve memory for the sprites, rather than using dedicated sprite RAM which must be set up by configuring. The best way of allocating memory from within Basic is by using DIM, for example:

```
DIM sp &1000
```

which reserves a block of memory &1000 bytes long and returns the address of the first byte in the variable *sp*. This area cannot easily be extended later, so make sure that it will hold all the sprites your program is likely to use. As a guide, each page of memory allocated using *CONFIGURE SpriteSize corresponds to &2000 (or 8K) bytes on a 300 series machine, and &8000 (or 32K) bytes on a 440.

Before storing sprites in our User area, we need to set up certain header information so that the operating system recognises it as a sprite area. This can be done using:

```
!sp=length     : REM Length of area
sp!4=0         : REM Initial sprite count
sp!8=16        : REM Sprite offset
sp!12=16       : REM Free space offset
```

The value of *length* should be the size of the area that has been reserved (&1000 in our example). All these values are updated by the operating system as sprites are manipulated, and once set up can be ignored.

USING THE SPRITE AREA

Having set up the User Sprite area, we can now turn our attention to manipulating sprites within it. As hinted at above, User Sprites are not handled using star commands, VDU statements or PLOT. Instead, an operating system SYS call, SYS "OS_SpriteOp" is used. The general form of this call is:

```
SYS "OS_SpriteOp",op,area,name,...
```

The value of *op* indicates the particular operation to be performed, for example loading a set of sprites or plotting a sprite. The parameter *area* is a pointer to our User Sprite area, and in our example would be the value of the variable *sp*. There is nothing to stop us having more than one User Sprite area at a time, in which case the value of *area* would indicate which one was being referred to. Next, *name* is normally a string that contains either the name of an individual sprite, or, in the case of load and save, a filename. Depending on the particular operation, there may be one or more additional parameters that have to be passed with the call.

The value of *op* is in fact made up of two components added together. The first of these is a number that represents the actual operation, while the second specifies how the sprites will be accessed. In our case, this second value will always be 256, which tells the operating system that our sprites are stored in a User area, and that we are accessing them using their names. The values of *op*, together with their functions, are given in the box below (with the 256 added in). Details of the other operations, and other types of access, can again be found in the *Programmer's Reference Manual.*

265	Clear all sprites
266	Load sprites
267	Merge sprites
268	Save sprites
270	Get sprite from screen
281	Delete sprite
290	Plot sprite on screen

Each of these is now described in detail. In each case it is assumed that *sp* contains a pointer to the start of our User Sprite area.

Clear All Sprites

SYS "OS_SpriteOp",265,sp
This simply clears all the sprites in our area, just as *SNEW does for the system sprite area.

Load Sprites

SYS "OS_SpriteOp",266,sp,filename$

Merge Sprites

SYS "OS_SpriteOp",267,sp,filename$

Save sprites

SYS "OS_SpriteOp",268,sp,filename$
These three calls perform the equivalents of *SLOAD, *SMERGE and *SSAVE on our User Sprite area. *filename$* can either be a string variable, a quoted string, or a pointer to the filename in memory. For example:

```
    $&A000="FileToLoad"
    SYS "OS_SpriteOp",266,sp,&A000
```

Get Sprite

SYS "OS_SpriteOp",270,sp,name$,flag
This performs a similar operation to *SGET or VDU23,27,1. That is it grabs the rectangle bounded by the last two graphics cursor positions, and makes it into a sprite. *name$* is the name given to the new sprite. The value of *flag* determines whether the current palette settings for the screen are stored with the sprite. If *flag* is zero then they will not be saved. However, if *flag* is one then the palette data is saved and can be recovered at a later stage.

Delete Sprite

SYS "OS_SpriteOp",281,sp,name$
This deletes the named sprite, in the same way as *SDELETE.

Plot Sprite

SYS "OS_SpriteOp",290,sp,name$,X,Y,mode
This plots the named sprite on the screen at position (X,Y). It is equivalent to:

```
    OSCLI "SCHOOSE "+name$
    PLOT &ED,X,Y
```

for system sprites.

Note that the operations of selecting the sprite and plotting it are combined in this single call. Whereas PLOT &ED etc. use the current GCOL plotting mode (AND, OR etc.), this call allows the plotting mode to be specified using the parameter *mode*. For example, setting *mode* to three would Exclusive-Or the sprite on to the screen.

The other sprite functions available using PLOT, for example plotting a sprite mask, can also be performed using SYS "OS_SpriteOp". Details of the necessary parameters can be found in the *Programmer's Reference Manual.*

A COMPLETE EXAMPLE

To clarify the calls explained above, the following section of code shows what is necessary to set up a &2000 byte sprite block, load a sprite file called 'MUSIC', and plot a sprite called 'TREBLE' in the middle of the screen. Finally, all the sprites in the area are cleared.

```
10 DIM sp &2000:!sp=&2000
20 sp!4=0:sp!8=16:sp!12=16
30 SYS "OS_SpriteOp",266,sp,"MUSIC"
40 SYS "OS_SpriteOp",290,sp,"TREBLE",
   640,512,0
50 SYS "OS_SpriteOp",265,sp
```

A sprite file called 'MUSIC' exists on the Archimedes Welcome Disc in the directory '$.ICONS', and this can be used to try out the above code.

USING USER SPRITES

Incorporating User Sprites in a program as it is being written should be fairly straightforward. The main point to remember is that a User Sprite area must be set up before any sprite operations are performed.

Converting an existing program that uses system sprites is normally also quite straightforward. The lines needed to define a sprite area are added at the start of the program, and all the operations are changed from star commands and PLOTs to the SYS calls described above. One thing to note is the difference in plotting sprites. Using the User Sprite calls it is not possible to select a sprite and then plot it repeatedly in different positions. Instead, the name of the sprite must be specified each time it is plotted.

Another point to consider is that once a User Sprite area has been set up it cannot be easily removed. Deleting all the sprites stored in the area will not free the allocated memory, though it could be used for other purposes, such as for storing a machine code program or a screen buffer.

Finally, there is an faster method for accessing User Sprites which involves referring to sprite definitions by their addresses in memory, rather than by their names. The *Programmer's Reference Manual* gives full details of this. RU

Leonardo

Archimedes Mode 12 Art Package

A comprehensive set of drawing options all available at four levels of magnification

Quick compressed screen files, so you can save many more screens on a disk

Available for all models of Archimedes

£17.50 inclusive

Leonardo 256

Archimedes Mode 15 Art Package

Mode 15 version of Leonardo with 256 colours plus a font generator

(Leonardo 256 requires 1 megabyte of memory)

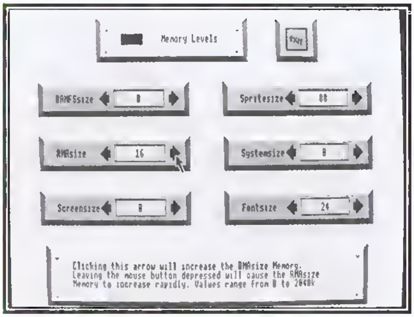
£19.50 inclusive

Available from : Beard Technology
111 Evering Road
London, N16 7SL.

LINGENUITY'S CONTROL PANEL

Configure your Archimedes using this icon driven control panel. David Spencer explains.

Control Panel from Lingenuity is a package that allows you to alter the configuration of your Archimedes by clicking on icons rather than by using a series of *CONFIGURE commands.



```
RAMFSsize  0      Spritesize  80
RMAsize    16     Systemsize  0
Screensize 0      Fontsize    24

Clicking this arrow will increase the RMAsize Memory.
Leaving the mouse button depressed will cause the RMAsize
Memory to increase rapidly. Values range from 0 to 2048k
```

Memory allocation menu

When you first boot the *Control Panel* disc, you are presented with the screen shown in the picture. Nine of the raised buttons represent certain classes of configuration, and clicking on these brings up a new screen allowing particular configuration settings to be changed. For example, clicking on the printer icon brings up a screen that allows the printer type and *ignore* character to be changed. The actual changes are made by clicking on left or right pointing arrows to step through the possible choices. Each screen includes an exit icon to return to the main screen. Clicking on the exit icon on the main screen will quit *Control Panel* and perform a hard reset to initialise all the new configurations. There is no way of exiting without executing a hard reset.

Configurations may be saved or reloaded by pressing the mouse's menu button when on the main screen, and selecting either load or save. This brings up a file dialogue box that allows you to move through directories before performing the operation.

Control Panel also features an on-line help system that constantly displays information about the icon that the mouse is currently pointing to.

An interesting feature of *Control Panel* is that the configuration files are saved as a series of *CONFIGURE commands. This means that to set up a particular saved configuration, you only need to type `*<filename>`.

Also provided on the *Control Panel* disc is a utility command to set a particular configuration from a file and then run an application. While this command can be quite useful, it does have a number of drawbacks. Firstly, once the application has been started, the BOOT configuration option is left set. Secondly, the application must be runnable from a !BOOT file in the root directory of the disc, and thirdly, there is no way of recovering the old configurations when the application has finished.

DOCUMENTATION

The documentation for the *Control Panel* is in the form of a four page booklet printed on five inch square paper, presumably to fit easily in the disc box. The instructions given fully cover the use of *Control Panel*, including its installation onto a hard disc using the supplied 'HDINSTALL' utility.

CONCLUSION

Control Panel is well presented, looks good when running, and does its job well. My only concern is the need for such a utility. It might well be useful until you get the hang of *CONFIGURE, but thereafter it is surely easier to type a couple of star commands. However, the ability to save a configuration as a text file is very nice, and well worth having. The price of £17.19 inc. VAT is reasonable when you consider what *Control Panel* offers. However, I must say that I would probably choose to spend five minutes learning about *CONFIGURE, rather than spending money on *Control Panel.*

Control Panel (£17.19 inc. VAT)
is supplied by Lingenuity,
P.O. Box 10, Halesworth, Suffolk IP19 0DX.
Tel. (098) 685 477

RU

ANNOUNCING A MAJOR NEW SOFTWARE RELEASE

Premier

THE FIRST OF A NEW GENERATION

Whether you want to write letters, create reports, edit Pascal or C files, send mail shots or write a novel, *PREMIER* will let you do it *YOUR* way. You can design your own screen layouts, define the action of ALL keys, and even invent your own commands .File size is limited only by your disc capacity, not your computer memory, and you may cut and paste between any number of files open at the same time. *PREMIER* will work interactively or from *JOB FILES,* to carry out *ANY* text processing task with the minimum of fuss.

This major new software release will be enhanced further to become *the* standard text processing package for the Archimedes. As a *RISC USER* reader you can take advantage of our special introductory offer and be in at the start. A demo disc is available for £5, refundable against our special fully inclusive purchase price of just £50, including free upgrades of all enhancements during 1989.

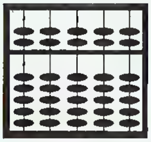
Circle Software
33,Restrop View,
Purton, Swindon,
Wilts. SN5 9DG

Extended Precision BASIC for the Archimedes

- ♦ Use all 27 floating point uniop and binops from BASIC.
  e.g. DVF division, LOG logarithm, ASN arcsine, POW power
- ♦ Get 19 significant figure precision in BASIC
- ♦ Use numbers from 10^-4931 to 10^+4931 or -10^4931
- ♦ Extended precision numbers are stored in BASIC strings

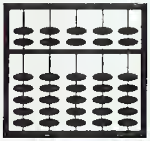
How does it work?

EP number strings and FP operations are passed to our BASIC function, FN_EpOp, as parameters. Acorn's floating point emulator or coprocessor is used to execute the FP operation. The value of the result is returned to BASIC as an EP numberstring. EP numbers returned from one operation can be passed directly to the next operation.



ABACUS TRAINING

29 Okus Grove, Upper Stratton,
Swindon, Wiltshire. SN2 6QA
Tel: (0793) 723347



Price: £30.00 including UK postage

MOVIE MAKER (Part 2)

by Lee Calcraft

Enhance your version of Movie Maker with a variety of sprite-based features.

This program needs a screen size of 160K, and a sprite size of up to 100K.

The Movie Maker program featured last month allows you to create animated sequences of all kinds with great ease. This month's listing provides a number of features that will make the process easier to perform, and more powerful in its implementation.

The new features allow you to pick up any rectangular area of the screen, and use this as a brush. This makes it very easy to *move* a person or object across the screen. Just draw the object once, and save the frame. Then pick it up, and re-draw it at its next position at the click of a button, before saving the next frame, and so on. But this is not all. The new features have a lot more to offer, and we will describe them in greater detail below.

But first, let's get the program running. The accompanying listing must be added to last month's program. To do this, load in last month's program (from either magazine or magazine disc). Then carefully type in the new additions. All but the first few lines follow on from the end of last month's program. Note by the way that no new lines have been inserted between existing lines, so that the program will not be altered by renumbering by tens, and in any case, the program makes no reference to any line number. When the work is complete, save the program to disc before running it.

When you run the new version of the program, you will see that there are three new options on the main menu:

```
     Sprt 1, Sprt 2 and L-Spr-S
```

The first two allow you to pick up and paint with any part of the screen. When you do this, the image is stored as a sprite (hence the name). The reason for having two menu entries, *Sprt* 1 and *Sprt 2*, is that it allows you to temporarily store two sprites. This can be very useful if you are trying to move two independent objects simultaneously. The third menu option allows you to load and save sprites.

If you have an object that you wish to move, press the *Select* button on either *Sprt 1* or *Sprt 2*. This will cause an empty rectangle to appear at the pointer. Move this to the bottom left-hand corner of the area that you wish to *grab*, and press *Select* again. Moving the mouse will now allow you to define the top-right-hand corner of the rubber-banded area, and pressing *Select* a third time will grab the image inside the rectangle.

The grabbed image will now move with the pointer, and at any time you can press *Select* to leave a copy of the image on the screen. Pressing *Adjust* will still save the current frame, as usual, even if the sprite image is still attached to the pointer. This makes it easy to create a sequence of images very quickly. Just press *Select* to put the image on screen, then *Adjust* to save the frame. Then move the pointer a couple of millimetres, and press *Select* to place the image at a new position, and then *Adjust* to save the new frame - and so on.

To disengage the image from the pointer, press *Menu*. This takes you out of Sprite mode, and you can use Movie Maker as normal. But the sprite image will be retained, and may be reinstated by pressing the *Menu* button, over the *Sprt 1* (or *Sprt 2)* menu box. To recap, pressing *Select* on *Sprt 1* or *Sprt 2* will clear any previously held sprite, ready to grab a new sprite from the screen, whereas pressing *Menu* on *Sprt 1* or *Sprt 2* will reinstate the last-used sprite. The *Sprt 2* option works in exactly the same way as *Sprt 1,* but the two sprites are completely independent.

The third new option allows you to save or load Movie Maker's two sprites. If you press *Select* on the "S" end of the *L-Spr-S* box you will be asked for a filename under which to save the two sprites. If you press *Select* over the "L" of the *L-Spr-S* box you will be asked for a filename for a sprite load.

In the simplest case, this just allows you to save a pair of sprites for subsequent re-loading at a later date. But the real power of these options lies in the ability to import images from other software. All you need to do is to create mode 13 sprites named:

```
        MovieSpr1
and     MovieSpr2
```

and to save them using:

```
        *SSAVE filename
```

One of the easiest ways to create suitable sprites is with the RISC User Pixel Editor (Volume 1, Issues 4 and 8) reproduced on the Volume One Special Disc. Alternatively, if you have a suitable mode 13 screen, you can make sprites from parts of it using the Sprite Grabber from RISC User Volume 1, Issue 10. Sprites transferred using Movie Maker's Sprite Load option can of course be used as the basis of objects to be animated. But they can also be used as backgrounds to a particular scene. In this case, you will need to set up the background before saving the first frame.

Remember that if you are tempted to scroll the background to achieve a particular effect, you will use masses of RAM, since the change from one frame to the next will be considerable. To minimise the use of RAM you need to keep the differences between consecutive frames as small as possible. As an indicator of the effectiveness of Movie Maker's Delta File system for saving screens, Roger Burg's excellent "Metamorph" animation supplied on last month's magazine disc contained over 600 frames, and yet was only 300K in length.

Finally, there is one small option that we have not yet covered. If you press the *Menu* rather than the *Select* button on the "L" of the *L-Spr-S* box, the program will try to incorporate sprites already held in the machine. Again they must be named:

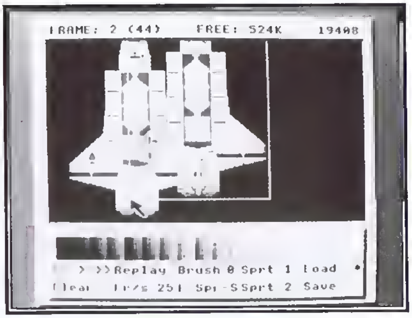
```
        MovieSpr1
and     MovieSpr2
```

This option is included just to save the bother of loading from disc sprites which may already be in the machine. It also allows you to go straight to Movie Maker after grabbing sprites from another piece of software. But you will of course need to use:

```
        *SRename oldname newname
```

if the sprites are not appropriately named.

This completes the features of Movie Maker. It is a very flexible piece of software, and from our experience it really does appeal to users of all ages. If you create any interesting sequences, please drop us a line. We hope to include some of them on future magazine discs. In the near future we will present a short program which will permit animated sequences from Movie Maker to be displayed in a stand-alone form, or from within the user's own software.



```
FRAME: 2 (44)      FREE: 524K      19408
  > >>Replay  Brush 0 Sprt  1  Load
Clear  Fr/s 25  Spr-SSprt  2  Save
```

Moving a sprite

Sample sprites are included on this month's magazine disc, together with the complete version of Movie Maker.

```
   10 REM            >AnimPt2
   20 REM Program    Mouse Animator
   30 REM Version    A 0.9K
   40 REM Author     Lee Calcraft
   50 REM RISC User  Jan/Feb 1989
   60 REM Copyright  Lee Calcraft
   70 :
   90 DIM buff &30,code &200,sname &10
  190 DATA << > >>,Replay,Brush,Sprt 1,L
oad  *
  200 DATA Clear,Fr/s,L-Spr-S,Sprt 2,Sav
e
  320  frameno=1:frametot=0:wx1%=48:wy1%
=48:wx2%=48:wy2%=48
  330  col=63:tint=192:spr1=FALSE:spr2=F
ALSE
  350  brush=0:file$="None Known":sfile$
=file$
```

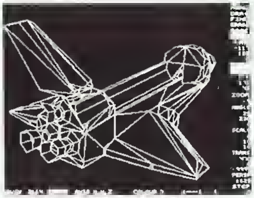
```
 1010  WHEN 4:IF z%=2 OR z%=4 PROCcopy(1
,z%=4)
 1050  WHEN 8:IF x%<640 THEN PROCsload E
LSE PROCssave
 1060  WHEN 9:IF z%=2 OR z%=4 PROCcopy(2
,z%=4)
 2460 PROCdialogue:tt%=OPENIN file$:ttt%
=EXT#tt%:CLOSE#tt%:IF ttt%>bsize% ERROR
255,"Not enough RAM" ELSE *FX200,1
 3580 :
 3590 DEFPROCcopy(sprno,newspr)
 3600 movequit=FALSE:SOUND 1,-10,70,1
 3610 sprite$="MovieSpr"+STR$sprno
 3620 PROCmousewait(0)
 3630 GCOL 3,63
 3640 IF newspr OR (sprno=1 AND NOT spr1
) OR (sprno=2 AND NOT spr2) THEN
 3650 REPEAT
 3660  MOUSE x%,y%,z%
 3670  RECTANGLE x%,y%,48,48
 3680  WAIT
 3690  RECTANGLE x%,y%,48,48
 3700 UNTIL z%=4 OR z%=2
 3710 IF z%=2 THEN movequit=TRUE
 3720 IF NOT movequit THEN
 3730  PROCmousewait(0)
 3740  bx%=x%:by%=y%
 3750  MOUSE TO x%+48,y%+48
 3760  REPEAT
 3770   MOUSE x%,y%,z%
 3780   RECTANGLE bx%,by%,x%-bx%,y%-by%
 3790   WAIT
 3800   RECTANGLE bx%,by%,x%-bx%,y%-by%
 3810  UNTIL z%=4 OR z%=2
 3820  IF z%=2 THEN movequit=TRUE
 3830 ENDIF
 3840 IF NOT movequit THEN
 3850  PROCmousewait(0)
 3860  wx%=x%-bx%:wy%=y%-by%
 3870  MOVE bx%,by%:MOVE x%,y%
 3880  IF sprno=1 THEN spr1=TRUE ELSE sp
r2=TRUE
 3890  OSCLI("SGET "+sprite$)
 3900  MOUSE TO bx%,by%
 3910 ENDIF
 3920 ELSE
 3930  IF sprno=1 THEN wx%=wx1%:wy%=wy1%
 ELSE wx%=wx2%:wy%=wy2%
 3940 ENDIF
 3950 IF NOT movequit THEN
 3960  OSCLI("SCHOOSE "+sprite$)
 3970  REPEAT
 3980   MOUSE x%,y%,z%
 3990   x%=4*(x% DIV 4):y%=4*(y% DIV 4)
 4000   RECTANGLE x%,y%,wx%,wy%
 4010   PLOT &ED,x%,y%
 4020   WAIT:WAIT
 4030   RECTANGLE x%,y%,wx%,wy%
 4040   PLOT &ED,x%,y%
 4050   IF z%=4 THEN GCOL 0,63:PLOT &ED,
x%,y%:GCOL 3,63
 4060   IF z%=1 THEN PROCmakeframe:GCOL
3,63:PROCheader:PROCmousewait(0)
 4070  UNTIL z%=2
 4080  IF sprno=1 THEN wx1%=wx%:wy1%=wy%
 ELSE wx2%=wx%:wy2%=wy%
 4090 ENDIF
 4100 GCOL 0,63
 4110 ENDPROC
 4120 :
 4130 DEFPROCssave
 4140 PROCdibox("Save Sprite")
 4150 PROCsdialogue:*FX200,1
 4160 OSCLI("SSAVE "+sfile$)
 4170 VDU26:PROCscreensetup(FALSE)
 4180 *FX200
 4190 ENDPROC
 4200 :
 4210 DEFPROCsload
 4220 IF z%=4 THEN
 4230  PROCdibox("Load Sprite")
 4240  PROCsdialogue:*FX200,1
 4250  OSCLI("LOAD "+sfile$)
 4260  VDU26:PROCscreensetup(FALSE)
 4270  *FX200
 4280 ELSE SOUND 1,-10,70,1
 4290 ENDIF
 4300 spr1=TRUE:spr2=TRUE
 4310 $sname="Moviespr1"
 4320 SYS "OS_SpriteOp",40,0,sname TO ,,
,wx1%,wy1%
 4330 wx1%=wx1%*4:wy1%=wy1%*4
 4340 $sname="Moviespr2"
 4350 SYS "OS_SpriteOp",40,0,sname TO ,,
,wx2%,wy2%
 4360 wx2%=wx2%*4:wy2%=wy2%*4
 4370 ENDPROC
 4380 :
 4390 DEFPROCsdialogue
 4400 PRINT'"Current name ";sfile$
 4410 INPUT'"Filename : "input$
 4420 IF input$<>"" THEN sfile$=input$
 4430 MOUSE ON:OFF
 4440 ENDPROC
```

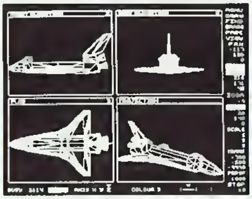
RU

SILICON VISION

SOFTWARE FOR THE ARCHIMEDES & BBC

SolidCAD





The ultimate 3D Draughting System for Architectural design, Interior design, Engineering Design and Teaching CDT. Allows drawing in plan, front & side elevations and also directly in 3D view. Includes powerful zoom & pan options for precision draughting and surface definition for creating solid colour objects. Also includes Sweep, Extrude & Macro facilities for designing very complex objects easily. Designs created with SolidCAD are compatible with the Realtime Graphics Language for high speed flicker-free animation. The custom Archimedes version also performs smooth shading for realism. SolidCAD(Arc) users can upgrade to the Realtime Solids Modeller (Arc) for £40.00.

£49.95 (ARC & BBC) New

REALTIME SOLIDS MODELLER

The package includes both the sophisticated design environment of SolidCAD and the high speed animation capability of a Realtime Graphics Language (RGL) module developed in pure ARM Risc code for supercharged performance. The package is ideal for Architectural design, Interior design, Engineering design & teaching CDT. The RGL module can be used to create standalone flicker-free animation of designs from your own programs. Smooth shading is also performed for realistic images. Through our in-house expertise in 3D Design and High-speed techniques, no other package can rival the design environment & animation speed of the Realtime Solids Modeller.

£89.95 (ARC) New

REALTIME GRAPHICS LANGUAGE

The Realtime Graphics Language rom provides a complete 3D Solids Wireframe animation system with 52 star commands and 3D Editors for designing objects to animate from your own programs. Includes a 35,000 pixels sec line generator for fast 3D drawing rates. 3D Rotate, Scale, Orbit, Perspective and Turtlegraphics. Also compatible with designs created with SolidCAD (BBC).

£49.95 (BBC)

SUPER-DUMP

The ultimate printer driver which takes advantage of the highest resolution capability of ordinary Epson compatible printers to provide 1920 × 1024 resolution. Images can also be scaled, positioned and previewed before printing. Fully compatible with SolidCAD, Realtime Graphics Language, Gate Array design system & 3D CAD Animation system. Your own graphics programs or other CAD packages can be made compatible with Super Dump by the addition of a few simple commands. An example program is included in the package.

£15.95 (BBC), £24.95 (ARC) New

Presentation Manager

[illegible] environment to create, edit and play back computer [illegible] and demonstrations. Also handles graph plotting for [illegible] which can be incorporated within the [illegible]

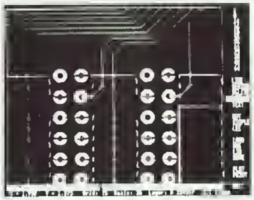
£34.95 (BBC), £49.95 (ARC) New

SILICON VISION LTD, SIGNAL HOUSE, LYON ROAD, HARROW MIDDLESEX HA1 2AG. TEL: 01-422 2274 or 01-861 2173 FAX: 01-427 5169. TELEX: 918266 SIGNAL G.

(Access/Mastercard/Eurocard accepted)

All prices include VAT and Carriage (Overseas orders add £4).

ARC-PCB



The ultimate PCB design system developed specifically for the Archimedes with a specification that cannot be matched. Includes Automatic routing, Rats-nesting, 8 layers, Surface mount capability, 0.001" resolution, 32" × 32" maximum board size, On-line Help, Fast Zoom Pan Redraw, Text & Silkscreen facility, Variable Line Pad Text Grid sizes, Part Libraries, Block Move Copy Rotate Mirror Erase options, and up to 300,000 components. For hardcopy, the system supports a large number of plotters and ordinary Epson compatible printers at very high resolutions (1920 × 1024) of 240 dots inch for near laser quality output. An entry level ARC-PCB system (without auto-routing) is available for £99.95 (Auto-routing upgrade £100). Enquire about our turnkey PCB design systems complete with Colour Archimedes and or plotters with prices from £1220 to £3909 (Inc VAT).

£195.00 (ARC) New

RiscBASIC

Supercharge your Archimedes Basic programs by compiling them automatically into pure ARM Risc code with the RiscBASIC compiler. Features include Relocatable modules, Cross reference of all variables, functions and procedures, Floating point and Integer support, Stand alone code generator, Optimising compiler & Full runtime error handler.

£99.95 (ARC) New

RiscFORTH

A new 32 bit implementation of the FORTH 83 standard, designed to take full advantage of the ARM architecture. Features include Multi-tasking, Optimising compiler, built in ARM assembler with floating point mnemonics, built in Full screen Editor, File system interface, OS calls support, Floating point & Integer maths, WIMP support, Single-step debugger, Shadow screen for documentation, Block manipulation, Dictionary & Vocabulary display, Call finding and a standalone code generator.

£99.95 (ARC). New

WIMP CHESS

This full-feature Chess package for the Arc is excellent value, as Stephen Streater discovers.

This complete chess package costs just £5.99, and consists of a single 3.5 inch floppy disc. It contains both program and instructions, thereby keeping overheads to a minimum. When the disc is booted you get a screen with a layout similar to Hearsay, the BEEBUG communications package, with a neat icon bar along the bottom of the screen, and the main work area containing any potential windows above it. During play, the main window contains a good 2D graphical representation of the board.

The icons along the bottom of the display consist of the following:

GET and **PUT** produce a file selector window, allowing you to save and load previous positions.

EDIT allows you to alter the state of the board to set up a particular position.

? gives a hint as to what move to play. The move is shown by flashing the borders of the source and destination square, which gives a clear indication as to your possible move.

PRINTER creates a disc file (suitable for printing) showing the course of the previous game, and includes the moves, score, depth of the computer search (user selectable at the start of the program).

* allows access to the OS. You will find this particularly useful for looking at the listing files.

The icon bar also displays the real time clock, and the time remaining for the current player (the real time clock got a bit messy as midnight passed, but otherwise worked ok).

Pressing the Menu button on the mouse will produce a list of further commands relevant to play. Those which I use most often include the option of a beep after each move, ARC v ARC for finishing off games (and which takes the start of the game from famous matches if you play from the start), and Undo which undoes a player's most recent move, and which can be used repeatedly.

Other features include "Switch sides", "Computer is white", "Computer is black", "Reverse", "Randomise", "Easy", "Level", "Depth", "New game" and "Save setup", which have their expected effects.

The program knows about castling and en passant, but the review copy can promote pawns only to queens, and can't un-queen pawns using "Undo Move". If you de-select the "Easy" option, the computer thinks in your time. A short experiment with the depth control didn't make any noticeable difference, so I left this at its default value of 29. The listing file claims depths of 2-4 for 1 second per move, and of 3-6 at an average 30 seconds per move, generally getting deeper as the game progresses. The deepest search (running overnight) was 10 ply.

Although my review copy was not the final release version, the only bugs I found were very minor ones which did not affect the playing, and which could be easily fixed. The program plays chess very well throughout the game, and with its vast range of levels, should give enjoyable games to players of all standards. This program is excellent value for money.

Product	*WIMP Chess*
Supplier	*David Pilling P.O. Box 22, Thornton Cleveleys, Blackpool FY5 1LR.*
Price	*£5.99 inc. p&p (no VAT)*

RU

ARCHIMEDES SOFTWARE

● Disc 1	EMACS Multiple buffer/dacument UNIX super editar with integral pragramming language, programs, tutarial, and manual
● Disc 2	Micra Spell 43,000 ward spell checker. Ideal far EMACS. New memary resident madule with cantinuaus spell check.
● Disc 3	Fartune Caakie. Database af aver 7000 amusing quatatians.
● Disc 4	XLISP Object arientated version af LISP with C saurce cade.
● Disc 5	C Taalkit. 20 programs including grep, awk, sed, ed, make, tail, crass ref., pretty print, file campare, sart, jain, split, etc.
● Disc 6	Kermit camms/file transfer program. Better than Acarn Kermit.
● Disc 7	Chess. A gaod chess game running under the Wimp enviranment. Lats af features; save & laad games, edit baard, computer play.
● Disc 8	Cross Star. Wimp based crossword puzzle solver with massive dictionary. Makes solving and compiling puzzles easy.
● Disc 9	File Tools. arc, compress, uuencode, atob, strings, cut, paste, fgrep.
● Disc 10	More C Tools. yacc & lex (with src. & examples), diff, ctags etc.

Each disc is £5.99 inc. Buy four claim one free!

**Available from David Pilling,
P.O. Box 22, Thornton Cleveleys, Blackpool. FY5 1LR.**

Archimedes Shareware Discs

Graphics Demo 1 (£3.50)

49 graphics demo programs plus MenuMaster–a program to allow you to edit the demo order and add your own favourite demos.

Shareware Disc Nº 1 (£3.50)

MenuMaster with seven more graphics demo programs plus Life, Mandelbrots, Structured Directory Lister, European Geography.

Shareware Disc Nº 2 (£4.50)

DFS reader, backup and archiver. 9 Graphics demos. Connect four, Mastermind, Solitaire and Star Trek games. 256 colour Sprite Editor. CMOS ram Editor, Disc Editor, LQ printer font definer, Matrix Functions, Memory Mappings & Vector Listings BASIC fast Screenload.

Save £1.50 – Buy all 3 for just £10.

Also Available...

WIMP Template Editor (£8)

Design your own windows, icons, menus and sprites – completely freehand in a wysiwyg format. Create WIMP templates which can easily be loaded from your own program – no more need for endless data blocks!!!!

Based on the original "Form Editor" program from Acorn Computers, modified and extended (with Acorn's permission!)

Prices include UK p&p. European prices as UK. Outside Europe sent by surface but add £1 per order for airmail. Pay by UK cheque or Eurocheque or credit our Girobank account 2341107. Sorry no credit cards.

Norwich Computer Services, 18 Mile End Road, Norwich, NR4 7QY. (0603–507057)

USING TWIN

Lee Calcraft shows how to get a bit more out of Acorn's Twin text editor.

Twin is an excellent product, and at around £30 inc. VAT it represents extremely good value, whether you are using it for Basic, C, Fortran, or whatever. The manual which accompanies it is well put together, but what follows is not in the manual!

MAKING IT INTO A MODULE

If you are a Twin-user without a hard disc, then you will know that you have to wait each time that Twin is called, while it is loaded from disc. You can avoid all this by using the Twin Module Maker (RISC User Volume 1 Issue 4). This creates a relocatable module which will download Twin instantaneously every time that it is called. This even avoids the necessity of having a disc version of Twin in your drive when the call is made.

USE OF RAM

Perhaps the most annoying thing about Twin is that it is not a Relocatable Module, and must be loaded (or downloaded) into a precisely located area of user RAM. This is fine if you have a 440 with 4 Mbytes of RAM, but with a 300 series machine, you can run into problems. The point is that when Basic calls Twin, it first creates an ASCII version of the resident program, and places it at the end of the current program. Twin then makes a copy of this in its own work area above itself in memory. This consumes a lot of RAM, and you can easily get into difficulties.

Roger Wilson (who wrote both Basic V and Twin) tells us that the way around the problem, if you are editing long programs, is to load Twin at &8000. This saves hundreds of K of RAM. But it will only work if the program you are editing is larger than 30 or 40K - otherwise the program will be overwritten. Ideally, you need two Modules containing Twin, one which will download it at &8000, and the other at say &40000 for shorter programs.

TWINO8

The best way to call Twin from Basic is with:

```
TWINO8
```

This strips off the line numbers before entering Twin, and makes editing much easier. It also makes returning to Basic much quicker if your program is of any length. Don't use TWINO9 onwards, because there is no easy way to remove the spaces that will be introduced into your listing. Of course, TWINO8 cannot be used if you use GOTO and the like, but no one with an Arc needs such things!

TEXT FILE ENTRY

If you place the following line on the boot file of your work disc:

```
*SET Alias$@RunType_FFF TWIN %*0
```

then if you execute the command:

```
*Name
```

on an ASCII file, Twin will automatically be called, and the file will be loaded into Twin for editing. This technique is particularly useful when using the RISC User ADFS menu, because by setting the *LoadType* on files of type FFD, FFE and FFF, you can load them into Twin directly from the Menu, simply by double-clicking the right-hand button on the file concerned (the right-hand button performs a load). For example:

```
*SET Alias$@LoadType_FFD TWIN %*0
```

FINDING YOUR PLACE

Use the *Go To Line* option (dividing the line number supplied by 10) to get to a known program line in Twin. Or alternatively, mark your program with easily searched for codes (e.g. REM **LO to indicate a Load routine). Then use option F4 (find and replace) to take you to **LO. RU

PipeDream

PipeDream is now available on the Acorn Archimedes. It provides comprehensive word processing, spreadsheet and database facilities integrated in a way only dreamed of before.

With other integrated packages, you have to divide your work into artificial sections, such as text, numbers and calculation, and database.

With PipeDream, you compose your work in the order you want to print it, with text and numbers all together in one document. Incorporate calculations directly into paragraphs of text and formatted paragraphs directly into spreadsheets.

PipeDream is ideal for all tasks involving words and numbers. From writing thank-you letters to encyclopaediae, invoices to cash flow forecasts, stamp-collection records to inventory management, film scripts to mail-shots.

PipeDream is 100% file and keystroke compatible with Z88 PipeDream and PipeDream on the IBM PC. It is also compatible with View Professional on the BBC microcomputer. You can create documents on any of these computers, transfer the files to any other and continue working, with no loss of information. No other software enables you to share your files with all these computers.

PipeDream for the Acorn Archimedes costs £99 + VAT.

It is not possible to detail all of PipeDream's facilities here. For full details or to order PipeDream call us on 0954 211472 or write to us at Colton Software, Broadway House, 149-151 St Neots Road, Cambridge CB3 7QJ.

PipeDream - power at your fingertips.

REVIEWS

Software Developer's Toolbox

Reviewed by David Spencer

The *Software Developer's Toolbox* from Acorn is a suite of utilities aimed at the serious programmer. By serious I mean those writing in high-level languages such as C or Fortran, or developing assembly language routines using Acorn's macro assembler, certainly not the average home user.

The *Software Developer's Toolbox* is supplied on two discs. The first of these contains twelve utilities, while the second contains the Symbolic Debugger. Both discs also include a copy of the floating point emulator, as this is needed by some of the utilities.

THE SYMBOLIC DEBUGGER

ASD, the Acorn Symbolic Debugger is the major component of the *Software Developer's Toolbox*. The purpose of ASD is to trace the execution of a compiled program written in a high-level language by relating this back to the source code. Using ASD it is possible to do such operations as printing out the names of variables as values are assigned to them. This would normally be impossible, because any reference to variable names is lost when the source code is compiled. ASD will work with code generated by the C, ISO-Pascal, or Fortran 77 compilers, although in each case it is necessary to compile the source code in a special way, and version 2 of the compiler is needed (see the News page in this issue).

UTILITIES

The other disc contains a bundle of twelve utilities, without even a menu to link them. Four of these are concerned with text files. These are:

COMMON which calculates the frequency of word usage in a file, or group of files, and then prints out the most common words.

DIFF to compare two text files. This produces as its output the steps needed to equate the two files. For example, it might state that a particular line has to be added in a certain place in the first file to make it match the second file.

GROPE to find a particular pattern in a file. In its simplest form this can be used to locate a single word. However, by using various wildcards similar to those in Twin, it is possible to find, say, all the labels in an assembly language listing.

WCto count the number of words, lines and characters in a file.

Another five utilities are devoted to the handling of Acorn Object Format (AOF), Acorn Library Format (ALF) and Acorn Image Format (AIF) files. AOF files are the object files produced by the language compilers and the assembler (OBJASM). ALF files are library files that can be shared between languages, and AIF files are executable machine code programs.

LINK combines AOF and ALF files to produce AIF files, and is essentially the same as the linker supplied with the language compilers.

AMU (A Make Utility) is a utility that is designed to make it easier to compile and link complete programs. However, the explanation in the manual of AMU is so horrendously complex that I would rather do the operation myself.

LIBFILE and OBJLIB are utilities for manipulating libraries in ALF file format. One possible use of them is to build up a standard set of functions that can then be used in a program written in any compiled language.

SQUEEZEis used to compress AIF files so that they take up less disc space. When a compressed file is executed, it is automatically expanded again.

DBUG is a major debugger utility which shouldn't be confused with the Symbolic Debugger, or the debugger module built into Arthur. The purpose of DBUG is to trace the execution of a machine code program. Once DBUG has been started, it is possible to load in an executable file, and perform operations such as single-stepping through the execution, and examining register contents.

SHELL is a Basic program which can be loaded using the LIBRARY or INSTALL commands. Its purpose is to allow other applications to be run from Basic, with control being returned afterwards to Basic rather than to Arthur.

MEMTEST, the final utility, simply tests the application memory in a number of different ways.

All the utilities in the *Software Developer's Toolbox* were written in C, and are therefore relatively long for the tasks they perform. Most of the utilities can take optional 'switches' to control their functions. These are given as '–<switch>' after the command. One particularly useful switch is '-help' to list help information for that utility. For example:

```
*COMMON -help
```

DOCUMENTATION

The documentation for the *Software Developer's Toolbox* comes in the form of two manuals. These are in the now familiar size and style used for all Archimedes software from Acorn. The first book, which is 44 pages long, details the use of the Symbolic Debugger. The second, which runs to 89 pages, explains all of the other utilities. Useful appendices are included on the linker file formats and Acorn's procedure call standard.

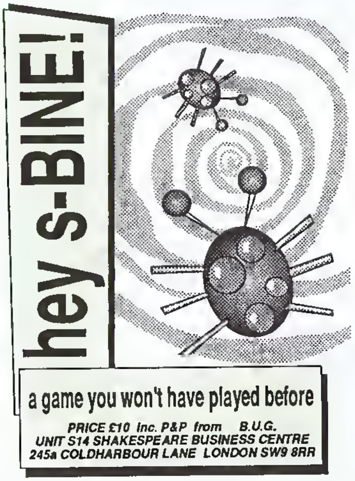
Also included are two reference cards, covering the ARM processor and the system star commands.

CONCLUSION

The *Software Developer's Toolbox* is one of the products that you either need or you don't. I suspect that the majority of Archimedes owners will not find a use for it, particularly at the high price charged by Acorn. Leaving this aside, I cannot fault the package, either in terms of software or documentation. However, when you consider what the *Software Developer's Toolbox* offers, I feel that a considerably lower price would be more realistic, and Acorn would probably sell more as a result.

Software Developer's Toolbox (£228.85 inc. VAT) is supplied by Acorn Computers Ltd., Fulbourn Road, Cambridge CB1 4JN. Tel. (0223) 245200

RU



PRICE £10 Inc. P&P from B.U.G.
UNIT S14 SHAKESPEARE BUSINESS CENTRE
245a COLDHARBOUR LANE LONDON SW9 8RR

GEOSCAN ©

A NEW GEOGRAPHY AND ECONOMICS PROGRAM ON WORLD DEVELOPMENT

WRITTEN SPECIALLY FOR THE ARCHIMEDES SERIES 310 & ABOVE

Simon Beaton B.Sc & Ron Dutton Ph.D

GEOSCAN consists of a single disc containing a large and varied database covering locational, physical, climatic and socio-economic information on 100 countries of the world. With this data there are program facilities for displaying the information in the form of text, statistics, graphs and diagrams, as well as for analysing and relating the data through rankings, correlations, scattergraphs, averages and mapped locations.

GEOSCAN has wide applicability as a straightforward and self-contained learning resource which will give students of all ages, valuable experience in the use of the computer for information retrieval. However, the comprehensive ASSIGNMENT PACK which accompanies the program has been written especially for students working at the level of GCSE and beyond. The content and the ready-to-duplicate nature of the worksheets will make them appeal particularly to teachers seeking Assignments for GCSE Geography and Economics.

An important feature of GEOSPAN is the annual up-date of its database as the latest statistics become available from international agencies. Each year, a new disc will be offered at very low cost, to the registered holders of the program , in exchange for their existing disc.

PRICE:£45.00 including VAT

Order direct, or send for Information leaflet to:

PASSKEY MARKETING
P.O. Box 649 Shenley Lodge Milton Keynes MK5 7AX

DIGITISING IN COLOUR

Lingenuity's latest interface turns the Watford Digitiser into a full colour system. Lee Calcraft reports.

In June 1988 we reviewed Watford Electronics' Video Digitiser. This unit will digitise monochrome pictures in real-time from sources such as a video camera or video recorder.

Lingenuity's new half-width podule is designed to be used in conjunction with the Watford unit to enable full-colour pictures to be captured. You therefore need to install in your machine both the Watford and the Lingenuity podules; though this is a very easy matter once you have a backplane fitted to your machine.

The Lingenuity board has just two BNC sockets at the rear. The "Video out" is connected via a short lead (supplied) to the "Video in" on the Watford Digitiser, while the "Video in" on the Lingenuity board is connected to the video source.

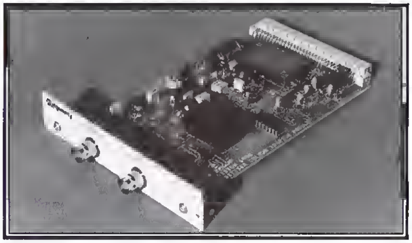
The Lingenuity unit works by separating the incoming video signal into its red, green and blue components, and sending them in sequence to the Watford Digitiser. The Lingenuity software causes the Watford Digitiser to grab the three resultant frames, and then combines these to form a mode 13 or mode 15 colour image.

As you will gather, this process cannot be instantaneous, and there is no question of grabbing a colour image in real time. In practice, you need an image which remains unchanged for around half a second. The system is thus unsuitable for grabbing action shots, unless you have some way of freezing the image. Some video recorders will produce a good enough freeze-frame image, though many cannot freeze a frame without introducing jitter or picture noise.

The software used to control the Colour Converter was written by Mike Harrison, who also wrote Watford's software, and it has some very nice touches. In particular there is a dithered mode, which uses colour mixing of adjacent pixels to provide a much wider effective colour palette (256,000 colours are claimed). The result of this is extremely good. Another feature enables you to adjust the colour make-up of a grabbed image. As you move the rectangular pointer around the screen, it changes colour indicating the colour bias which can be applied to the picture.

Now for the negative points. Although the software utilities supplied are of a high standard, the user interface is not. When you boot the software, you are given very little on-screen help if your machine is not appropriately configured (you need 600K of user RAM), and when the program runs, the start-up screen consists of nothing more than a grey rectangle one centimetre square. The user menu is crudely drawn. It is not under WIMP control, and sits annoyingly in the centre of the screen. If your video source is not supplying a signal, the program just beeps at you, with no message whatsoever. Finally, you cannot use the "Save Sprite" options because the program will not run unless you set sprite space to 1 or less (as instructed in the manual).

The manual also leaves much to be desired. It is just 10 pages of type-written A5, with a slide binding which makes it difficult to read the left hand side of each page. Because of the way in which it is organised and presented I found it hard to find things in it without reading it from cover to cover each time.



As to the board itself, it is well made, and performs faultlessly. It is a pity that so little attention has been invested in the finish of the product - that is to say the software user interface and the manual: especially in view of the relatively high price of this product.

Colour Converter (£195.44 inc. VAT and p&p) is supplied by Lingenuity, PO Box 10, Holesworth, Suffolk. IP19 0DX. Tel. (098) 685 477

RU

ARE YOU GOOD ENOUGH?

As the leaders in saftware far the Archimedes range af camputers, CLARES MICRO SUPPLIES are laaking ta extend aur range even further. We are laaking for peaple who are as excited by the Archimedes as we are.

If you have written any pragrams, completed ar not, then we wauld like ta hear from you.

If yau have any ideas far pragrams and have the ability to execute the ideas then we want ta hear from you.

If yau have the ability ta pragram the Archimedes but nat the ideas to pragram then we want ta hear from yau.

Pragrams can be written in any language as long as they perform their stated task. Many af aur programs contain large chunks af BASIC with ARM cade in the areas that it is needed. BASIC on the Archimedes is a very pawerful language and we da nat attach any snab value ta its use. If yaur pragram daes what is meant ta da then thats all we are interested in. Why not jain the tap team an the Archimedes. Yau get the support of aur in-hause team, privileged access thraugh us ta Acarn and invitations ta our infarmal pragrammers seminars.

The most impartant paint is that yau will be earning tap royalty rates of if yau prefer we will purchase yaur pragram outright.

Please write, in canfidence, ta Mr. D. Clare at:

Clares Micro Supplies,
98 Middlewich Road,
Northwich,
CHESHIRE CW9 7DA

If yau have a program either camplete or in develapment then please enclase a copy for our evaluation.

Ta protect yourself we advise that yau lodge a capy of the program with yaur bank or salicitor BEFORE you send us a copy. You can then prave that yaur program pre-dates anything that we have.

Act taday and become part af the leading software team producing saftware for the warlds fastest micro.

RISC USER TOOLBOX (7)

by David Spencer

Four new filing system utilities for the RISC User Toolbox.

There are four new commands this month. The first two, *CATALL and *EXALL, perform a similar function to *CAT and *EX, except that they work recursively. This means that the contents of any sub-directories are also listed, as are their sub-directories and so on. The other two commands, *FIND and *GOTO, will search through all directories and sub-directories for files which match the wild-carded name given in the command.

ENTERING THE LISTING

Start by adding the listing given here to the existing Toolbox source code. This is exactly as in previous months. Save the complete source code, and run it to assemble and save the Toolbox module. Type QUIT followed by TOOLBOX to install the module.

*CATALL AND *EXALL

The syntax for these commands is:

```
*CATALL [<pathname>] [I]   and
*EXALL [<pathname>] [I]
```

The parameter <pathname> specifies at which directory to start. If none is given, then the current directory is used. To list all the files, use a pathname of $. For example:

```
*CATALL $
```

will catalogue all the files, and sub-directories, on the current disc. You can also specify a filing system and drive number in the pathname, for example:

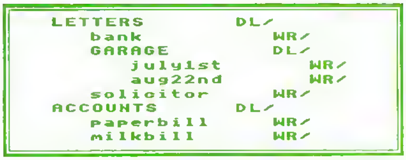
```
*EXALL ADFS::0.$.PROGRAMS
```

The optional 'I' parameter determines the format of the display. Without the 'I', each filename is listed as the full pathname from the starting directory, for example:

```
letters.view.doctor
```

while with the 'I', only the actual filename is displayed, this being indented according to its directory level. A typical display in this format, which is particularly useful when mapping an entire disc for reference, is shown in figure 1. In both cases, directory names are shown in reverse text.

If you want to give the 'I' parameter without specifying a starting directory, then the optional pathname must be given as "", rather than just being omitted. Otherwise, the 'I' will be taken as the starting directory.



```
LETTERS               DL/
    bank                  WR/
    GARAGE                DL/
        julyist               WR/
        aug22nd               WR/
    solicitor             WR/
ACCOUNTS              DL/
    paperbill             WR/
    milkbill              WR/
```

Figure 1. The indented display format

*FIND AND *GOTO

These have the syntax:

```
*FIND <filename> [<pathname>] [D] [F]
*GOTO <filename> [<pathname>] [D]
```

The given filename can include the # and * wildcards. For example:

```
*FIND Tool* $
```

will search through all the files on the disc (the starting directory being $) for those beginning with the letters 'Tool'.

*FIND simply lists the pathname (relative to the current directory) of any files that match the given name. *GOTO on the other hand will select as the current directory the first directory that contains a file that matches the name. For example:

```
*GOTO Myfile $
```

would leave you in the first directory that contains a file called 'Silly'.

Both *FIND and *GOTO can take an optional 'D' parameter. For example:

```
*FIND Tool* $ D
```

If the D is present, then directory names are included in the search, while if it is absent only files are considered. Therefore, the above command with the 'D' would match a directory called 'Toolbox', while without the 'D' it wouldn't. The same comments about not giving the pathname in the command apply to these commands as they did to *CATALL and *EXALL.

*FIND can also take an optional 'F' which controls the display format. Without the 'F', only the pathname and attributes of matched files are listed. With the 'F', the full file information is displayed.

```
   80 DIM code 15000
  347 EQUS "Catall":EQUB 0
  348 ALIGN:EQUD catac:EQUD &20000
  349 EQUD catasyn:EQUD catahlp
  350 EQUS "Exall":EQUB 0
  351 ALIGN:EQUD exac:EQUD &20000
  352 EQUD exasyn:EQUD exahlp
  353 EQUS "Find":EQUB 0
  354 ALIGN:EQUD findc:EQUD &40001
  355 EQUD findsyn:EQUD findhlp
  356 EQUS "Goto":EQUB 0
  357 ALIGN:EQUD gotoc:EQUD &30001
  358 EQUD gotosyn:EQUD gotohlp
 1413 .catahlp EQUS "*Catall lists all t
he files in the directory hierarchy star
ting from the specified directory.":EQUB
 13
 1423 .catasyn EQUS "Syntax: Catall [<pa
thname>] [I]":EQUB 0:ALIGN
 1433 .exahlp EQUS "*Exall displays info
rmation on all the files in the director
y hierarchy starting from the specified
directory.":EQUB 13
 1443 .exasyn EQUS "Syntax: Exall [<path
name>] [I]":EQUB 0:ALIGN
 1444 .findhlp EQUS "*Find displays the
pathname of any file matching the specif
ied name.":EQUB 13
 1445 .findsyn EQUS "Syntax: Find <filen
ame> [<pathname>] [D] [F]":EQUB 0:ALIGN
 1446 .gotohlp EQUS "*Goto moves to the
directory containing the specified file.
":EQUB 13
 1447 .gotosyn EQUS "Syntax: Goto <filen
ame> [<pathname>] [D]":EQUB 0:ALIGN
 2661 EQUS "HuntFile":EQUB 0
 2662 EQUS "FileInfo":EQUB 0
 2663 EQUS "WildMatch":EQUB 0
 3715 B swil1:B swil2:B swil3
13580 .catac MOV R2,#0:B exac2
13590 .exac MOV R2,#1
13600 .exac2 STMFD R13!,{R14}
13610 LDR R12,[R12]:STR R2,[R12,#16]
13620 CMP R1,#2:MOV R3,#0:BNE exac3
13630 STMFD R13!,{R0-R2}:BL fopt
13640 CMP R1,#ASC"I":LDMFD R13!,{R0-R2}
13650 MOVEQ R3,#1:BEQ exac3
13660 .bqual ADR R0,bqe:LDMFD R13!,{R14}
13670 ORRS PC,R14,#1<<28
13680 .bqe EQUD 0:EQUS "Bad option to To
olbox command":EQUB 0:ALIGN
13690 .exac3 ADR R1,ecrout
13700 STR R3,[R12,#32]:BL swil1
13710 BL couprt:LDMFD R13!,{PC}^
13720 .swil2 STMFD R13!,{R0-R4,R14}
13730 LDRB R3,[R1]:CMP R3,#ASC"."
13740 ADDEQ R1,R1,#1:LDRB R3,[R1]
13750 CMP R3,#0:BEQ noi:CMP R2,#1
13760 BEQ ind:LDR R4,[R0,#16]:CMP R4,#2
13770 BLEQ rev:MOV R0,R1:SWI "OS_Write0"
13780 SWI &12E:CMP R4,#2:BLEQ rev:B noi
13790 .ind LDRB R2,[R1],#1:CMP R2,#0
13800 SWIEQ &120:SWIEQ &120:BEQ noi
13810 CMP R2,#ASC".":SWIEQ &120
13820 SWIEQ &120:B ind
13830 .noi LDR R0,[R13]:LDR R4,[R0,#16]
13840 CMP R4,#2:BLEQ rev:ADD R0,R0,#20
13850 MOV R2,#0:.fnp LDRB R3,[R0,R2]
13860 CMP R3,#0:ADDNE R2,R2,#1:BNE fnp
13870 SWI "OS_Write0":.fnp2 CMP R2,#11
13880 BEQ fnp3:SWI &120:ADD R2,R2,#1
13890 B fnp2
13900 .fnp3 LDR R0,[R13]:ADD R0,R0,#12
13910 MOV R3,#0:LDRB R2,[R0,#4]
13920 CMP R2,#2:SWIEQ &144:MOVEQ R3,#1
13930 LDRB R2,[R0]:TST R2,#8
13940 SWINE &14C:ADDNE R3,R3,#1
13950 TST R2,#2:SWINE &157:ADDNE R3,R3,#
1:TST R2,#1:SWINE &152:ADDNE R3,R3,#1
13960 SWI &12F
13970 TST R2,#32:SWINE &157:ADDNE R3,R3,
#1:TST R2,#16:SWINE &152:ADDNE R3,R3,#1
13980 .atpad CMP R3,#6:SWINE &120
13990 ADDNE R3,R3,#1:BNE atpad
14000 LDR R0,[R13,#12]:CMP R0,#0
14010 BEQ findone
14020 LDR R0,[R13]:LDR R1,[R0]
14030 LDR R2,mask:AND R1,R1,R2
14040 CMP R1,R2:BEQ stmpd
14050 SWI "OS_WriteS"
14060 EQUS "              ":EQUB 0
14070 LDR R1,[R0],#4:BL adrprt
14080 SWI &120:LDR R1,[R0]:BL adrprt
14090 B dolen
14100 .stmpd LDR R1,[R0]
14110 LDR R3,[R0,#16]:CMP R3,#2
14120 LDREQ R2,dirtxt:LDREQ R3,dirtxt+4
14130 BEQ dirtype
14140 AND R1,R1,R2,LSR #12
14150 MOV R2,R1,LSR #8
14160 MOV R0,#18:SWI "OS_FSControl"
14170 .dirtype MOV R1,#4:MOV R0,R2
14180 .lotype SWI "OS_WriteC"
14190 MOV R0,R0,LSR #8:SUBS R1,R1,#1
14200 BNE lotype
14210 MOV R1,#4:MOV R0,R3
14220 .hitype SWI "OS_WriteC"
14230 MOV R0,R0,LSR #8:SUBS R1,R1,#1
14240 BNE hitype:SWI &120:LDR R0,[R13]
14250 LDR R2,[R0],#4:STR R2,[R12,#4]
14260 LDR R2,[R0]:STR R2,[R12]
14270 MOV R0,R12:ADD R1,R12,#64
14280 MOV R2,#100:ADR R3,tiform
14290 SWI "OS_ConvertDateAndTime"
14300 SWI "OS_Write0"
14310 .dolen SWI &120:LDR R0,[R13]
14320 LDR R1,[R0,#8]:BL adrprt
14330 .findone LDR R0,[R13]
```

```
14340 LDR R4,[R0,#16]:CMP R4,#2
14350 BLEQ rev:SWI "OS_NewLine"
14360 LDMFD R13!,{R0-R4,PC}
14370 .mask EQUD &FFF00000
14380 .tiform EQUS "%24:%MI:%SE %DY-%M3-
%CE%YR":EQUB 0:ALIGN
14390 .dirtxt EQUS "dir     "
14400 .ecrout LDR R2,[R12,#32]
14410 LDR R3,[R12,#16]:BL swil2
14420 LDR R0,[R12,#20]:ADD R0,R0,#1
14430 STR R0,[R12,#20]:LDMFD R13!,{PC}
14440 .couprt STMFD R13!,{R14}
14450 LDR R0,[R12,#20]:ADD R1,R12,#64:MO
V R2,#64:SWI "OS_ConvertCardinal4"
14460 SWI "OS_Write0":SWI "OS_WriteS"
14470 EQUS " object(s) found.":EQUW &D0A
14480 EQUB 0:LDMFD R13!,{PC}
14490 .swil1 STMFD R13!,{R0-R7,R14}
14500 ADD R3,R12,#256:MOV R2,#1<<29
14510 STR R1,[R12,#24]:SWI "OS_GSInit"
14520 .ncl SWI "OS_GSRead":MOVCS R1,#0
14530 STRB R1,[R3],#1:BCC ncl
14540 SUB R7,R3,#1:STR R1,[R12,#20]
14550 STR R7,[R12,#28]:BL rec
14560 LDMFD R13!,{R0-R7,PC}
14570 .rec STMFD R13!,{R14}
14580 MOV R4,#0:ADD R1,R12,#256
14590 .rec2 ADD R2,R12,#128:MOV R3,#1
14600 MOV R5,#128:ADR R6,wildname
14610 MOV R0,#10:SWI "OS_GBPB"
14620 CMP R3,#0:LDMEQFD R13!,{PC}
14630 STMFD R13!,{R0-R7}:ADR R0,recb
14640 STMFD R13!,{R0}:ADD R0,R12,#128
14650 LDR R1,[R12,#28]:LDR PC,[R12,#24]
14660 .recb LDMFD R13!,{R0-R7}
14670 LDR R5,[R2,#16]:CMP R5,#2
14680 BNE rec2:STMFD R13!,{R4,R7}
14690 ADD R4,R12,#256:CMP R4,R7
14700 LDRNEB R4,[R7,#-1]:CMPNE R4,#&3A
14710 CMPNE R4,#ASC"-"
14720 MOVNE R0,#ASC".":STRNEB R0,[R7],#1
14730 ADD R2,R2,#20:.ccl LDRB R0,[R2],#1
14740 STRB R0,[R7],#1:CMP R0,#0:BNE ccl
14750 SUB R7,R7,#1:BL rec:MOV R0,#0
14760 LDMFD R13!,{R4,R7}:STRB R0,[R7]
14770 MOV R0,#0:STRB R0,[R7]:B rec2
14780 .wildname EQUS "*":EQUB 0
14790 .adrprt STMFD R13!,{R0,R14}
14800 MOV R0,R1:ADD R1,R12,#64
14810 MOV R2,#9:SWI "OS_ConvertHex8"
14820 SWI "OS_Write0"
14830 LDMFD R13!,{R0,PC}
14840 .swil3 STMFD R13!,{R14}
14850 .noeol LDRB R2,[R1],#1:CMP R2,#ASC
"*":BEQ mwild
14860 LDRB R3,[R0],#1:CMP R2,#ASC"#"
14870 BEQ mat:EOR R3,R2,R3
14880 AND R3,R3,#&DF:CMP R3,#0
14890 .mat MOVNE R3,#1:MOVEQ R3,#0
14900 LDMNEFD R13!,{PC}^:CMP R2,#0
14910 BLNE swil3:LDMFD R13!,{PC}^
14920 .mwild STMFD R13!,{R0-R1}
14930 BL swil3:CMP R3,#0
14940 LDMFD R13!,{R0-R1}
14950 LDMEQFD R13!,{PC}^:LDRB R2,[R0]
14960 CMP R2,#0:MOVEQ R3,#1
14970 LDMEQFD R13!,{PC}^:ADD R0,R0,#1
14980 B mwild
14990 .fopt STMFD R13!,{R14}
15000 MOV R2,#1<<29:SWI "OS_GSInit"
15010 .fopt2 SWI "OS_GSRead":BCC fopt2
15020 .fopt3 CMP R1,#ASC" "
15030 LDREQB R1,[R0],#1:BEQ fopt3
15040 AND R1,R1,#&DF:LDMFD R13!,{PC}
15050 .findc MOV R2,#0:B gotoc2
15060 .gotoc MOV R2,#1
15070 .gotoc2 STMFD R13!,{R14}:MOV R4,R1
15080 LDR R12,[R12]:STR R2,[R12,#16]
15090 MOV R2,#1<<29:SWI "OS_GSInit"
15100 STR R13,[R12,#36]:ADD R3,R12,#512
15110 .gtoc3 SWI "OS_GSRead":MOVCS R1,#0
15120 STRB R1,[R3],#1:BCC gtoc3
15130 SUB R0,R0,#1:STMFD R13!,{R0}
15140 CMP R4,#3:MOV R5,#0:BCC gtoc4
15150 BL fopt
15160 CMP R1,#ASC"D":ORREQ R5,R5,#2
15170 BNE gt35:.gtoc34 LDRB R1,[R0],#1
15180 CMP R1,#ASC" ":BEQ gtoc34
15190 BCC gtoc4
15200 AND R1,R1,#&DF:.gt35 CMP R1,#&46
15210 LDMNEFD R13!,{R0}:BNE bqual
15220 ORR R5,R5,#1
15230 .gtoc4 STR R5,[R12,#32]
15240 LDMFD R13!,{R0}
15250 ADR R1,fgrout:BL swil1
15260 BL couprt:LDMFD R13!,{PC}^
15270 .fgrout STMFD R13!,{R0-R1}
15280 LDR R3,[R0,#16]:CMP R3,#2
15290 LDREQ R3,[R12,#32]:ANDEQ R3,R3,#2
15300 CMPEQ R3,#0:LDMEQFD R13!,{R0-R1,PC
}:ADD R0,R0,#20:ADD R1,R12,#512
15310 BL swil3:LDMFD R13!,{R0-R1}
15320 CMP R3,#0:LDMNEFD R13!,{PC}
15330 LDR R3,[R12,#16]:CMP R3,#1:BEQ gr
15340 LDR R3,[R12,#20]:ADD R3,R3,#1
15350 STR R3,[R12,#20]:MOV R2,#0
15360 LDR R3,[R12,#32]:AND R3,R3,#1
15370 BL swil2:LDMFD R13!,{PC}
15380 .gr LDR R1,dirc:STR R1,[R12,#252]
15390 ADD R0,R12,#256:SWI &122
15400 SWI "OS_Write0":SWI &122
15410 SWI "OS_NewLine"
15420 ADD R0,R12,#252:SWI "OS_CLI"
15430 LDR R13,[R12,#36]:LDMFD R13!,{PC}
15440 .dirc EQUS "DIR "
15450 .rev STMFD R13!,{R0,R14}:SWI &117
15460 SWI &111:SWI &105:MOV R0,#7
15470 BL sndnull:LDMFD R13!,{R0,PC}
```

BACKUP IN BROMLEY

AUTHORISED ACORN DEALERS

We offer full technical backup, service and advice as well as a comprehensive range of software and hardware for your Archimedes. If you have a problem - we can help you.

Call in for a software or hardware demonstration, or phone for prices and availability

Data Store

6 Chatterton Road, Bromley, Kent
Telephone : 01-460 8991 (closed Wednesdays)

Archimedes Macro Assembler Version 2

The Wingpass Archimedes Macro Assembler takes ARM assembly language source files and produces Acorn Object Format (AOF) files suitable for linking with other AOF files produced by a high level language compiler.

Features:

* Accepts the full ARM instruction set including Floating Point instructions. All FP data types (Float, Double, Extended and Packed) are supported.

* Allows 'C' language header files to be included so definitions of structures, unions and #defines need only be done once.

* Powerful macro facility includes parameter specification by position or by name, parameter defaults and variable length parameter lists.

* Example files supplied include: routines calling and called from 'C' routines obeying the calling standard, example relocatable module, complete Mandelbrot set program.

£ 49.95 (inc. VAT and P&P)

Wingpass Ltd, 19 Lincoln Ave, Twickenham, MIDDX TW2 6NH

INTRODUCING ARM ASSEMBLER (9)

by Lee Calcraft

BASIC'S INBUILT ASSEMBLER

In this final instalment of the series, I want to take a closer look at Basic V's inbuilt assembler. It is very similar in concept to the 6502 assembler which forms an integral part of BBC Basic on the Model B and Master Series computers, and readers familiar with the Beeb's assembler will probably find that most of what follows is already second nature to them.

Perhaps the best way to describe its features is to look at a simple example of its use. Listing 1 contains an assembler program which will display the letter "A". Line 50 sets aside an area of RAM in which the program will be assembled. We have reserved &1000 bytes, or 4K. This is enough room for 1024 ARM instructions, since each instruction is four bytes wide.

Listing 1

```
   10 REM >ARM91
   20 REM Assembler Demo
   30 REM by Lee Calcraft
   40 :
   50 DIM space &1000
   60 FOR pass=0 TO 1
   70 P%=space
   80 [
   90 OPT pass*3
  100 .start
  110 SWI 256+ASC"A"
  120 MOV PC,R14;        Return
  130 :
  140 ]
  150 NEXT
```

ASSEMBLER PARAPHERNALIA

On the Basic V assembler (just as with earlier BBC Basic assemblers for the 6502 language), all assembled code must be enclosed in a FOR-NEXT loop to achieve so-called two-pass assembly. This ensures that all forward references are correctly catered for. Within the FOR loop, and before the start of the ARM code itself, there are three vital ingredients. Firstly the variable P% must be assigned to the start address of the area of RAM reserved for assembler use. In our case, P% is set to the value held in the variable *space*, giving the start address of the block of RAM reserved in line 50. The assembler uses P% to keep the address of the next instruction to be assembled. And as assembly takes place, P% is automatically updated. To verify this, you could print out the value of P% before and after assembly. The two resultant values will differ by a factor of four times the number of instructions in the simplest case.

After the initial assignment of P%, there must be an open square bracket to indicate that all which follows forms part of the assembly listing. This is an important dividing line. You must not place Basic instructions after this point, and no assembler mnemonics should appear before it. At the end of the listing, you will see that there is a corresponding closing bracket (at line 140), followed by a NEXT statement, to complete the listing.

The last of the three essentials before the start of the code itself is the OPT directive. OPT is an assembler directive which must be followed by a number between 0 and 7. The number is used to specify the way in which assembly errors are to be handled, whether a listing is to be displayed during assembly, and whether so-called *offset assembly* is to be used (see later). Figure 1 gives a table of these options.

Value	Offset	Errors	Listing
0	No	No	No
1	No	No	Yes
2	No	Yes	No
3	No	Yes	Yes
4	Yes	No	No
5	Yes	No	Yes
6	Yes	Yes	No
7	Yes	Yes	Yes

Figure 1. The effect of OPT

In our short program we have set OPT to the value pass*3, and since the variable *pass* takes the value zero on the first pass, and one on the second, OPT will be set first to zero, and then to three. This means that on the first pass, no errors will be signalled, and no listing will be given, while on the second pass, we will have error notification and an assembler listing. The reason why we have set it to zero for the first pass, is that there will be a number of errors on the first pass as the assembler attempts to use labels for which it does not as yet have an address. By the second pass, it will have read or calculated the values of all labels, and no errors should occur. If they do, then they are real errors that we must correct in our listing.

Finally, we come to the code itself. This just displays the letter "A" on the screen, and returns to Basic. Note though, the way in which Basic functions, such as ASC(), can be incorporated within an assembly listing. You can even use trig functions such as SIN or TAN, or even the GET function. But remember, the calculations made as a result are only made at *assembly time*, not at *run time*.

```
00008FD8
00008FD8
00008FD8                       OPT pass*3
00008FD8              .start
00008FD8  EF000141             SWI 256+ASC"A"
00008FDC  E1A0F00E             MOV PC,R14;      Return
00008FE0
00008FE0
```

Figure 2. Output from the Arc's Basic Assembler

If you now enter the program in listing 1, and run it, you will see the assembly listing displayed on the screen as the program is assembled (see figure 2). Such a display can be extremely instructive. It contains four fields. From left to right these are:

The instruction address in hex
The instruction word
Labels
The instruction mnemonic

If you take a look at figure 1 you will see that the label *start* and the SWI instruction have the same address (&8FD8). The following instruction (MOV PC,R14 - which executes a return to Basic) is at &8FDC. This is four bytes higher in memory than the first instruction, since all ARM instructions are four bytes wide.

CALL AND USR

Even though you have typed RUN, the machine code program will not be executed. All that will happen is that it will be assembled into the designated area of RAM. To execute the code, you will need to use the CALL statement or the USR function. If you type:

```
    CALL start
```
or `z=USR(start)`
you will see the letter "A" appear on the screen, indicating a successful execution of the program.

The main difference between CALL and USR is that USR allows a value to be returned from the machine code routine. This is always the contents of ARM register R0 at the time of exit. Thus if you issue:

```
    z=USR(start)
```
Then the machine code routine at address *start* will be called, and on completion, the variable z will hold the final contents of R0.

Both CALL and USR also permit values to be assigned at the start of execution. Whenever code is executed using CALL or USR, the contents of Basic variables A% to H% are loaded into ARM registers R0 to R7. Additionally, the CALL statement can take a number of optional parameters separated by commas. In this case R9 will be set up to point to the list of parameters, and R10 to the number of parameters supplied. For further details, see *ARM Assembly Language Programming*, page 92. Parameters cannot be issued with USR.

SUPPLYING DATA

The four assembler directives EQUB, EQUW, EQUD and EQUS permit the assembly language programmer to supply data with a piece of machine code. EQUB supplies a byte, EQUW at two-byte word, EQUD a four byte (32-bit) word, and EQUS a string. The program in listing 2 illustrates the use of these directives. If you run it, you will see that it prints the string "Some Text", and below it, the hex number &FFFFFFFF.

Listing 2

```
 10 REM >ARM92
 20 REM Assembler Demo 2
 30 REM by Lee Calcraft
 40 :
 50 DIM space &1000
 60 FOR pass=0 TO 1
 70 P%=space
 80 [
 90 OPT pass*3
100 .start
110 SWI "OS_WriteS"
120 EQUS "Some Text"
130 EQUW &0D0A
140 EQUB 0
150 ALIGN
160 LDR R0,data
170 MOV PC,R14;        Return
180 :
190  data
200 EQUD &FFFFFFFF
210 :
220 ]
230 NEXT
240 :
250 PRINT~USR(start)
```

The string is generated using OS_WriteS, which sends to the screen the string of bytes immediately following the instruction, and terminated with a zero byte (see part 2 of this series). We have used:

```
EQUS "Some Text"
```

to insert into memory the sequence of ASCII characters in the supplied string. This is followed by:

```
EQUW &0D0A
```

This places the two bytes &0A and &0D into RAM, to give a Carriage Return and Line Feed. We have then used:

```
EQUB 0
```

to place a zero byte at the end of the sequence. This tells OS_WriteS that the string is complete.

You will see that we have followed this by an ALIGN directive. This simply aligns the next instruction to a 32-bit word boundary. This is essential, because the ARM expects all instructions to be aligned in this way. The instruction following the ALIGN directive is:

```
LDR R0,data
```

This loads register R0 with the 32 bit word at address *data*. Then the routine ends with a return to Basic. The 32 bit word is supplied by the EQUD directive at line 200, and this results in the value &FFFFFFFF being loaded into R0 immediately prior to returning to Basic. Because the final statement in the program is:

```
PRINT ~USR(start)
```

the hex value &FFFFFFFF is sent to the screen.

OFFSET ASSEMBLY

When assembly is carried out using values of OPT between 4 and 7, so-called *offset assembly* is employed. In this case the code is still assembled as if it were to be executed from the address given by P%. But it is not stored at this address. It is stored at the address held in the variable O%; and O% is automatically incremented by the assembler in the same way as P%. There are a number of occasions when offset assembly is useful. The most notable is in the assembly of code for a relocatable module. This must be assembled in a completely relocatable manner, and the most convenient way to do this is to use an assembly start address of zero.

Obviously we cannot assemble the code directly at 0000, since this would totally corrupt the system stack. But by using offset assembly, and setting P% to 0, and O% to a reserved area of user RAM, we can achieve this with great ease. The code will be assembled as if it was to run at 0000, but it will actually be placed at a convenient place in user RAM, from where it may be directly saved to disc as a relocatable module. Of course, when it is used it will be loaded into the Relocatable Module Area, at an address determined by the Module Manager.

All too soon we have come to the end of our allotted space, and I hope that you have found this series useful in getting to grips with ARM assembler. Even now, some 18 months after the release of the Archimedes, the ARM is still the most powerful processor available on any microcomputer. Because of its native speed, and the power and versatility of its instruction set, the ARM is a joy to program, and I hope that this short series has given you more than a hint of its capabilities. RU

Dabhand Users: New for 89!

NEW: ABC Version 2: Now even better!

"...Excellent Dabs Press product. Buy it!" RISC User Dec.88

"ABC is a vital part of any programmer's toolbox, it puts compilers on other systems to shame. Unquestionably one of the most impressive pieces of software I have yet seen running on the Archimedes." A&B Dec 1988

"...I can tell you now, I am very impressed. This is a superb package." Archive Dec 1988

The above quotes were refering to Version 1 of ABC - ABC Version 2 is even better! Version 2 allows use ot double and extended precision floating point, multiple exits from procedures and functions, RETURN parameter passing, new compiler directives and very much more.

As all reviewers have found, ABC makes writing machine code programs and relocatable modules easy. Programs can be written using the full range ot BASIC V's error messages and report's, and once fully working, run through ABC which transtorms them into machine code, ready tor immediate action.

Demo Disc: We have produced a demo disc of ABC which still supports over 100 commands. It costs just £2 and this is retundable on any subsequent purchase. A full specitication sheet is also available on request.

ABC was written by Paul Fellows head of the team that wrote the Archimedes Operating System. Registered users will receive version 2 automatically tree ot charge - how's that tor service!

£99.95

NEW: Archimedes Operating System

Hot off the presses, this latest Dabhand Guide is essential reading for every user of the Operating System including Arthur *and* RISC OS.

Its 320 pages are packed with vital information and covers topics such as VIDC, MEMC, IOC, Sound, SWIs, Vectors, Filing System and very much more. Price is just **£14.95** or **£21.95** with programs disc and manual.

Archimedes Assembly Language

386 pages devoted to programming the Archimedes in machine code. At **£14.95** it is the *only* book which deals specifically with assembler on the Archimedes. Book and programs disc ordered together – **£21.95.**

Arcendium: Board Game Fun for All!

Backgammon, Draughts, Reversi and Ouadline – four games tor the price of one! Sure to get the whole tamily using the Archimedes.

Arcendium allows numerous levels ot play either against an opponent or the computer. Sound, speech and superb colour graphics all add to the fun. Includes on-line help and re-play facilities plus more... **£14.95**

Alerion: Archimedes Arcade Action!

The highly acclaimed all-action shoot-em-up for the Archimedes. 256 colour mode graphics, with digitised speech. Impossible to tinish! Superb fun!

"...the gameplay makes this game a winner...a first rate game..." A&B Computing.

£14.95

Archimedes PC Emulator Shareware

Five discs tull ot PC software tested with the emulator to ensure compatibility. The collection includes software you would normally expect to pay a fortune tor and includes a wordprocessor, spreadsheet, games, tlowchart designer, printer utilities, and very much more.

Mindreader is a powerful wordprocessor with mailmerge, spelling checker, and an amazing 'word anticipate' feature. As Easy As is a tull-tunction Lotus-compatible spreadsheet, with macros and graphics. Games include Bridge, colour chess, plane-flying etc. Utilities include a writing style analyser, font designer, letter-quality text printer ... and so the list goes on. Nearly 4Mbs ot PC action complete with User Guide. **£34.95**

Dabhand Guides for your Archimedes

In reviewing our books PCW said this: *"...a truly professional publication."*

C: A Dabhand Guide

PCW said: *"I only wish this book had been available when I was learning C."* It you want to learn C on your Archimedes then this 512 page volume is the way to do it. Full details on ANSI C and includes Archimedes specific text.

At **£14.95** it represents quite incredible value. Book and programs disc **£21.95.** Essential for all users of C on the Archimedes.

FREE on Request Bumper Catalogue!

DABS PRESS

5 Victoria Lane (RUF), Whitefield, Manchester M25 6AL

Tel: 061-766 8423 BT Gold: 72:MAG11596 Prestel: 942B7621D

Prices include VAT and P&P (UK/BFPO/CI). **ACCESS/VISA** accepted by post/phone/Mailbox/ in person. Cheques and POs to address above. Dabs Press products available from all good dealers. Add £2.50 (£12 air) if outside UK. Official orders welcome.

HINTS & TIPS HINTS & TIPS

Rounded up by Lee Calcraft.

HIGH RES MODES WITHOUT A MULTI-SYNC

by W.T.Over

As indicated in the VDU Planning Sheet Generator article in RISC User Volume 1 Issue 6, you can make use of the Arc's high resolution modes without a multi-sync monitor by dumping the results to a printer.

Use:

```
   *Configure MonitorType n
```

where n=1 for high resolution use, or n=0 an ordinary monitor. Of course, you cannot see anything on the screen in high res. modes if you haven't got a multi-sync monitor, but screen dumps to printer (or disc) will work perfectly.

Fortunately, you do not even need to perform a reset after reconfiguring the monitor type. This means that you can have a program which displays your graphics on a non-high res. screen, then when all is ok it issues a *con.mon.1, redraws the screen in a high res. mode, and then performs *HardCopyFx 1 1 1 0 1 (or whatever). When the printout is complete, just set *Con.Mon.0, and it is as simple as that.

I have used this method to generate customised graph paper in mode 18, giving a resolution of 640 by 512.

DIY ERRORS

by Lee Calcraft

Basic V has many useful extensions. The new statement:

```
   ERROR number,string
```

is simple, yet extremely powerful. It allows you to incorporate your own error conditions within a Basic program (or indeed by using the OS counterpart *ERROR, it can be used in machine code and other programs).

To illustrate its power, suppose you have a program with its own error trapping, in which you would like to incorporate a special error trap in case the user attempts to load a data file which is too long for the amount of reserved RAM. The following line is all you need:

```
   IF filelen>maxlen THEN ERROR 255,"This
file is too long"
```

If the length of the file is too long, Basic will generate a special error, with error number 255, and with associated message: "This file is too long".

The beauty of it is that the error will be handled in the normal way by your own error handler, and you no longer need to write a complete handling routine for every special error condition that you wish to trap.

The only thing to watch is that you do not assign error numbers already used for other purposes. In practice, you could give all your user-errors the same error number (eg. 255).

PRESERVING C FLAGS

by David Pilling

When writing a routine using OBJASM for use from inside compiled C programs, it is important that all flags are preserved on exit from your routine. If you don't do this, your program may appear to work, but will occasionally demonstrate baffling bugs. So, your code should be bracketed by two statements like this:

```
   STMFD R12!,{R14}
   .
   .
   LDMFD R12!,{R15}^
```

The important bit is the circumflex character which preserves the flags.

RISC OS OR ARTHUR?

by David Spencer

As with earlier Acorn machines, you can use INKEY(-256) to discover which machine version you have. Arthur 1.2 gives 160, while the new RISC OS operating system gives 161.

To try it out, use:

```
   PRINT INKEY(-256)
```

DUAL SCREEN BAD MODE

by Lee Calcraft

After using certain software, you may be surprised to find that you cannot use the more memory intensive screen modes, even when your machine is correctly configured. If you get a Bad Mode message when trying to execute a MODE 15 statement, say, when you have a full 160K of screen RAM configured, try executing:

```
   *FX114,1
```

This will reset the computer to single screen-bank working, and will allow the whole of screen RAM to be used for any given screen mode as appropriate. RU

Postbag

We welcome your letters for publication in our Postbag page on all matters related to RISC User and the Archimedes.

RISC OS DOUBTS

In the review of RISC OS (RISC User Volume 2 Issue 1) it mentions the addition of a RAM filing system. Does this require the purchase of an additional unit, or does it use the RAM already in the machine?

Secondly, I purchased the PC Emulator software. Will RISC OS affect the way this runs? Will I need to get an update? Furthermore, I ordered the Programmer's Reference Manual when I bought my Archimedes. Does the new RISC OS render that two-volume book useless?

R.H.Bentall

The RAM filing system does not require the purchase of any additional RAM. The user simply allocates part of the system's existing RAM to create a RAM filing system when required.

Acorn has striven to make RISC OS compatible with programs which run under Arthur 1.2. Although there is nothing to stop Acorn producing a new version of the PC Emulator at some stage, the existing one should work fine.

The Programmer's Reference Manual will certainly not be useless - almost everything in these tomes remains both correct and relevant. However, there are new features in RISC OS for which technical information at the level of the Reference Manual will certainly be needed. Acorn says that a new edition of the Programmer's Reference Manual will be produced for April, and that arrangements for users with the first edition are currently being considered.

AUTOSKETCH REVIEW

I read the review of AutoSketch (RISC User Volume 1 Issue 10) and wonder whether the reviewer is entirely familiar with the PC. Perhaps the most well-known CAD program for the PC is AutoCad by AutoDesk (the publishers of AutoSketch), and it is in use by many companies in this country. It also supports all sorts of hardware and software add-ons. As a single user program it is quite amazing to use, even to an Arc fan. It costs about £2500!

So AutoDesk released a cheap, sawn-off version of AutoCad called AutoSketch costing around £75 for the PC. It is a useful introduction to computer-aided drawing, easing the novice into the world of CAD, but it is not a full-blown CAD package. In fact, the extra facilities the reviewer sought are but a very small subset of AutoCad.

Perhaps the most important point is that a major software house has recognised the potential of the Arc. If AutoDesk issue AutoCad on the Arc, Acorn are going to sell a lot more computers.

Garry Orford

Mr Orford implies criticism of some of my more adverse comments regarding AutoSketch. I fully agree with his point about major software houses supporting the Archimedes. I believe this is essential for the longer term success of Acorn, and thus AutoDesk deserve undoubted praise. What I did try to say was that I thought the Arc was capable of much more than was offered by AutoSketch in the field of CAD, and if AutoCad appears for this machine as well, then I and many others I am sure will be only too pleased, particularly if it costs less than £2500. Mike Williams.

COMPUTER SPEAK

I find that there is a trend in RISC User to drift into the usual *Computer Speak*, i.e. jargon. For instance, when writing about file handling, we are told a channel is allocated .... and will be called a handle. Why not a channel, number or name?

We all know that buzz words are used to distinguished between experts and newcomers. As an example, I have still not seen a satisfactory explanation of vectors in my non-computer language.

E.R.Brealey

I feel that Mr Brealey is a little unfair. Nearly every field of activity has associated with it a specialised vocabulary. This is quite reasonable, and leads to precision and conciseness in communication. We see no reason not to use the correct technical word in RISC User when appropriate, particularly where Acorn have set a precedent. However, we do lay considerable stress on ensuring clarity of explanation, particularly where an article is aimed at less technical readers, and we do try to ensure that any new terms (such as handle) are explained adequately. RU

RISC User Magazine Disc

January/February 1989

FULL SCREEN ZOOM AND PAN A module to allow screen images to be zoomed and panned. A demonstration is also included.

MOVIE MAKER
The complete Movie Maker system including a sample sprite file.

RISC USER TOOLBOX
The commands *CATALL, *EXALL, *FIND and *GOTO are added to the Toolbox.

USER SPRITES
A demonstration program to show how user sprites can be incorporated in programs.

ARCHIMEDES VISUALS
Two fractal-based programs which draw a fractal tree, and an entire fractal landscape.

INTRODUCING ARM ASSEMBLER
Two short programs to illustrate the use of Basic's inbuilt assembler.

BONUS ITEMS

KEYSTRIP GENERATOR A handy program to print out high quality user-defined keystrips on Epson compatible printers.

SOLITAIRE AND CODEBREAKER A colourful implementation of two well known logic games.

PROARTISAN SCREENS Some sample screens to show what can be achieved with Clare's ProArtisan.

SOLIDCAD DEMO A sample screen to demonstrate the smooth shading effects available with this new package from Silicon Vision.

ARCSCAN DATA Bibliographies for this issue of RISC User and the latest BEEBUG (Vol.7 No.7) to include in *Arcscan*.

RISC User magazine discs are available to order, or by subscription. Full details of prices etc. are given on the back cover of each issue of RISC User.

DESKTOP DIARY UTILITY

Make your "Archimedes Desktop Diaries" convenient to use.

See a full year at a glance.
A single screen shows part of the entry for every day of the year. See the free days or spot that elusive entry at a glance.

Instant viewing of the full entry for any day.
Skim the mouse over the year to scan the full entries for any day, & just 'click' on a day to make an entry.

Print all of a month with just one command.
Select the month, or months, to be printed. Then choose from either a full page, weekly formatted print-out, or a quick single line per day, showing only those days which have an entry.

Easy to use, includes Help & *commands.

£9.95
(incl. vat & U.K. p&p)

Cheque or P.O. to Official Orders also welcome.

QUERCUS COMPUTER SYSTEMS
Courtyard House, North Otterington,
N.Yorks. DL7 9EP

QUERCUS specialises in customised technical & process control software, on a wide variety of micros. Tel. 0609 70643 to discuss your requirements.

Software Project Coordinator

An interesting position has arisen in our software development department for someone to coordinate various new software projects on the Archimedes.

Reporting to the software manager, the successful applicant will have the opportunity to control software projects from conception right through to release of the completed packages. This is not a programming position, but knowledge of Basic and Assembler is essential (and C preferable) together with familiarity with popular applications software. The ability to write clearly will also be necessary to produce comprehensive program specifications and user documentation.

If you are interested in this challenging job, please apply in writing to the Personnel Manager at the address below.

BEEBUG Ltd. Dolphin Place, Holywell Hill,
St. Albans, Herts. AL1 1EX

RISC USER magazine

MEMBERSHIP

RISC User is available only on subscription at the rates shown below. Full subscribers to RISC User may also take out a reduced rate subscription to BEEBUG (the magazine for the BBC micro and Master series).

All subscriptions, including overseas, should be in pounds sterling. We will also accept payment by Connect, Access and Visa, and official UK orders are welcome.

RISC USER SUBSCRIPTION RATES		RISC USER & BEEBUG
£14.50	1 year (10 issues) UK, BFPO, Ch.I	£23.00
£20.00	Rest of Europe & Eire	£33.00
£25.00	Middle East	£40.00
£27.00	Americas & Africa	£44.00
£29.00	Elsewhere	£48.00

BACK ISSUES

We intend to maintain stocks of back issues New subscribers can therefore obtain earlier copies to provide a complete set from Vol.1 Issue 1. Back issues cost £1.20 each. You should also include postage as shown:

Destination	UK, BFPO, Ch.Is	Europe plus Eire	Elsewhere
First Issue	60p	£1	£2
Each subsequent Issue	30p	50p	£1

MAGAZINE DISC

The programs from each issue of RISC User are available on a monthly 3.5" disc. This will be available to order, or you may take out a subscription to ensure that the disc arrives at the same time as the magazine. The first issue (with six programs and animated graphics demo) is at the special low price of £3.75. The disc for each issue contains all the programs from the magazine, together with a number of additional items by way of demonstration, all at the standard rate of £4.75.

MAGAZINE DISC PRICES	UK	Overseas
Single Issue discs	£ 4.75	£ 4.75
Six months subscription	£25.50	£30.00
Twelve months subscription	£50.00	£56.00

Disc subscriptions include postage, but you should add 60p per disc for individual orders (30p for additional discs on the same order).

All orders, subscriptions and other correspondence should be addressed to:

RISC User, Dolphin Place, Holywell Hill, St Albans, Herts AL1 1EX.
Telephone: St Albans (0727) 40303
(24hrs answerphone service for payment by Connect, Access or Visa card)